# Sistema de diagnóstico de queratocono basado en inferencia bayesiana.

**Bustamante Juárez Eduardo**[1]

[1]*Centro de Investigación en Ciencias. Universidad Autónoma del Estado del Morelos.*

Este manuscrito fue compilado el 12 de junio de 2024

**Abstract**

El presente reporte ilustra el proyecto “Sistema de diagnóstico de queratocono basado en inferencia bayesiana” aplicable al tercer parcial del curso “Búsqueda de Soluciones e Inferencia Bayesiana” del programa educativo Licenciatura en Inteligencia Artificial, adscrito al Centro de Investigación en Ciencias de la Universidad Autónoma del Estado de Morelos.
El proyecto propuesto consiste en desarrollar un clasificador bayesiano para el diagnóstico de queratocono, a partir de mapas topográficos corneales (obtenidas en formato de imagen) etiquetadas como pacientes con queratocono, pacientes sospechosos de queratocono y pacientes con ojos normales. La inferencia bayesiana es emplea con el fin de modelar la probabilidad de la presencia de queratocono con base en las características de las imágenes. Los resultados preliminares muestran una precisión del XX% en la detección de queratocono, lo que sugiere la viabilidad de utilizar este enfoque para el diagnóstico temprano de la enfermedad.

**Keywords:** *Clasificador Bayesiano, Queratocono, Inferencia Bayesiana, mapas topográficos corneales, Diagnóstico*

**Corresponding author:** Bustamante Juárez Eduardo. *E-mail address:* eduardo.bustamantej@uaem.edu.mx
**Received:** Junio 12, 2024

## Contents



## 1. Introducción.

El queratocono es una enfermedad corneal progresiva caracterizada por un adelgazamiento irregular y elevación de la córnea en forma de cono. Este cambio en la forma de la córnea puede provocar una pérdida importante de visión y, en casos graves, requiere un trasplante de córnea. La detección temprana del queratocono es importante para iniciar tratamientos que puedan retardar o detener la progresión del queratocono, mejorando así la calidad de vida del paciente. En la práctica clínica, el diagnóstico de queratocono se realiza principalmente mediante el análisis de mapas topográficos de la córnea, que proporcionan información detallada sobre la curvatura y la forma de la córnea.

A pesar de la importancia del diagnóstico temprano del queratocono, la interpretación manual de los mapas de topografía corneal puede ser subjetiva y depender de la experiencia del médico. Además, con la creciente cantidad de datos en la práctica clínica, existe la necesidad de métodos automatizados y precisos que puedan ayudar a los profesionales sanitarios a diagnosticar esta enfermedad. El propósito de este proyecto es desarrollar un sistema automatizado basado en la inferencia bayesiana para clasificar los mapas de topografía corneal en tres categorías: ojos normales, ojos con sospecha de queratocono y ojos con queratocono.

Para realizar este proyecto se utilizaron datos de imágenes de topografía corneal adquiridas con un instrumento Pentacam. Un oftalmólogo recopiló y etiquetó las imágenes para garantizar la alta calidad y precisión de las etiquetas. Estos datos incluyen siete mapas corneales diferentes por ojo, que proporcionan información detallada sobre diversos aspectos de la curvatura y la forma de la córnea. En total, el conjunto de datos constaba de 3794 imágenes de desarrollo y 1050 imágenes de prueba, todas redimensionadas a 224 x 224 píxeles para facilitar el procesamiento.

Los datos utilizados en este proyecto son de gran relevancia clínica ya que reflejan la situación real en la práctica oftalmológica. Estos datos permitirán el desarrollo y evaluación de un modelo que potencialmente podría usarse en la práctica clínica para mejorar el diagnóstico del queratocono. Además, el uso de imágenes de alta calidad etiquetadas por expertos garantiza que el modelo aprenda características significativas que son importantes para diagnosticar enfermedades con precisión.

El clasificador gaussiano Naive Bayes es una técnica de aprendizaje automático basada en el teorema de Bayes, que supone que las características son independientes entre sí. A pesar de su simplicidad, ha demostrado ser eficaz en una variedad de tareas de clasificación. En este proyecto se utiliza el clasificador Naive Bayes-Gauss debido a su capacidad para manejar datos continuos y su eficiencia computacional.

El análisis de componentes principales (PCA) es una técnica de reducción de dimensionalidad que transforma datos en un nuevo espacio de características de dimensiones inferiores y maximiza la varianza explicada por cada componente. Esto no sólo reduce la

complejidad del modelo sino que también mejora el rendimiento al eliminar el ruido y la redundancia en los datos.

Las técnicas avanzadas de aprendizaje automático, como Naive Bayes Gaussian y PCA, combinadas con datos clínicos de alta calidad, permiten el desarrollo de sistemas de diagnóstico de queratocono automatizados eficientes y precisos. Este sistema tiene el potencial de ayudar a los profesionales de la salud en la detección temprana y el tratamiento del queratocono, mejorando así los resultados clínicos de los pacientes.

## 2. Objetivos.

### 2.1. Objetivos Generales.

Generar un "Sistema de diagnóstico de queratocono basado en Inferencia bayesiana" para clasificar a los pacientes en tres categorías: paciente con queratocono, paciente sospechoso de queratocono y paciente con ojos normales; utilizando mapas topográficos corneales.

### 2.2. Objetivos Específicos.

1. Analizar y clasificar los mapas topográficos corneales de los pacientes a través de la implementación de un clasificador bayesiano.
2. Con base en los mapas topográficos corneales contenidas en el conjunto de datos etiquetados de los pacientes con queratocono, pacientes sospechosos de queratocono y pacientes normales; entrenar el clasificador bayesiano.
3. Modelar por medio de la inferencia bayesiana y con base en las características obtenidas de los mapas topográficos corneales, la probabilidad de que un paciente presente queratocono.
4. Evaluar la precisión del "Sistema de diagnóstico de queratocono basado en Inferencia bayesiana" en la detección de queratocono y comparar los resultados con otras técnicas.

## 3. Metodología.

A continuación, se describe el conjunto de técnicas, herramientas y procedimientos empleados durante el desarrollo del proyecto "Sistema de diagnóstico de queratocono basado en inferencia bayesiana". La información siguiente detalla la estrategia a seguir durante el desarrollo sistemático y organizado del proyecto, con el propósito de alcanzar los objetivos planteados.

### 3.1. Adquisición de Datos.

Los conjunto de mapas topográficos corneales a utilizar en el presente proyecto, fueron obtenidos del artículo "A Hybrid Deep Learning Construct for Detecting Keratoconus From Corneal Maps", publicado en Translational Vision Science and Technology (2021). Los datos serán utilizados bajo la licencia Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

El conjunto de entrenamiento almacena 2961 mapas topográficos corneales de 423 casos. Para el conjunto de validación del modelo, se cuenta con 1050 mapas topográficos corneales de 150 casos. Cada caso contiene 7 mapas topográficos corneales, cuáles corresponden a la excentricidad anterior y posterior, elevación anterior y posterior, curvatura sagital anterior y posterior, y mapas de espesor corneal para extraer rasgos de la córnea profunda.

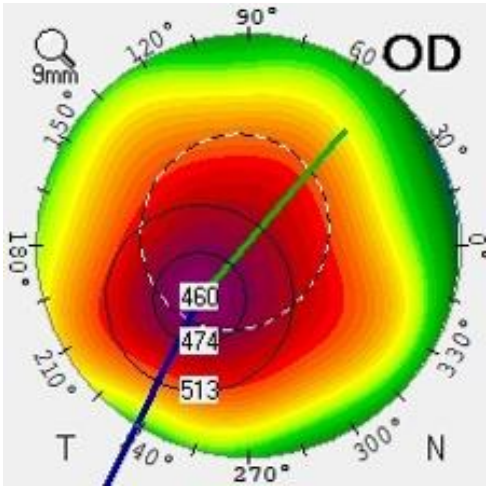


**Figure 1.** Mapa topográfico corneal CT_A.

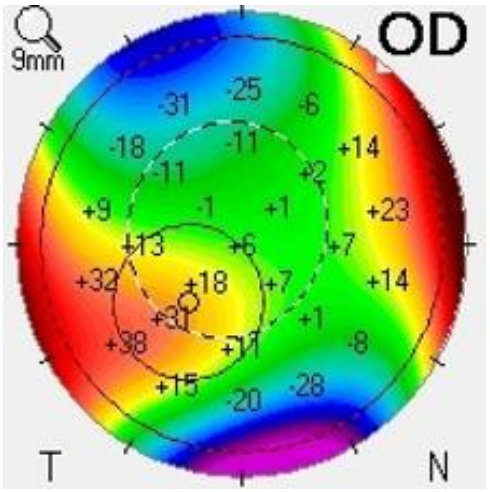


**Figure 2.** Mapa topográfico corneal EC_A.

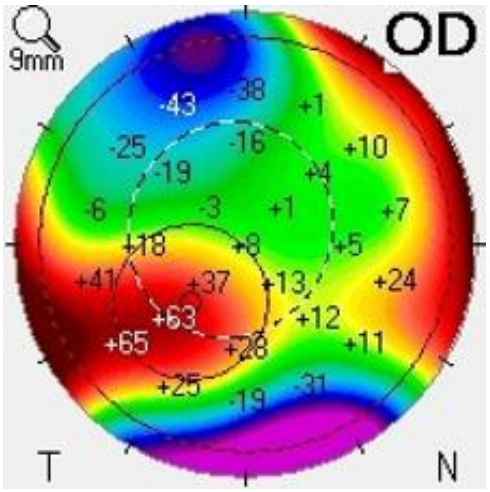


**Figure 3.** Mapa topográfico corneal EC_P.

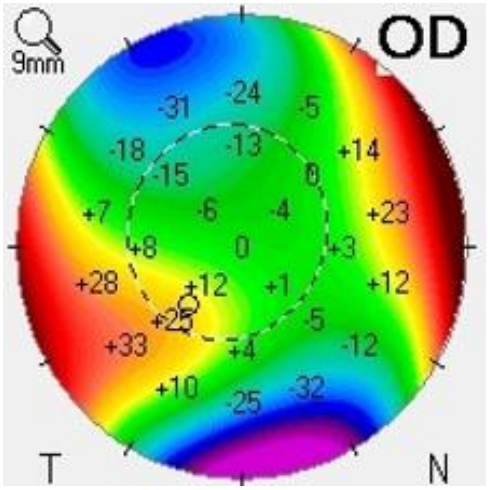


**Figure 4.** Mapa topográfico corneal Elv_A.

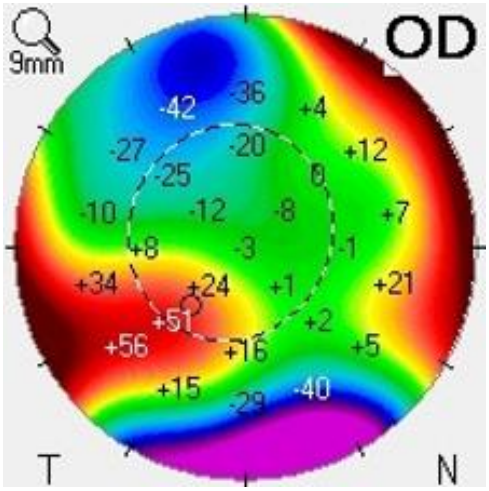


**Figure 5.** Mapa topográfico corneal Elv_P.

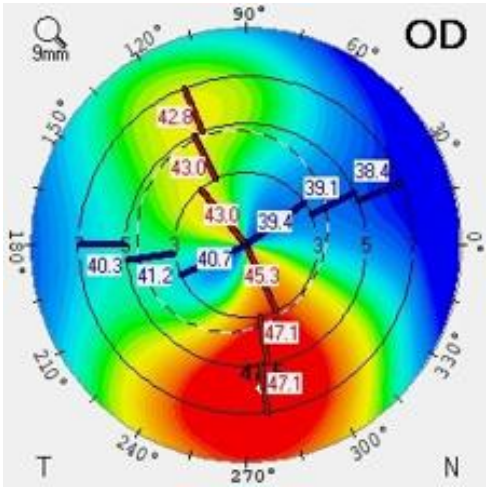


**Figure 6.** Mapa topográfico corneal Sag_A.

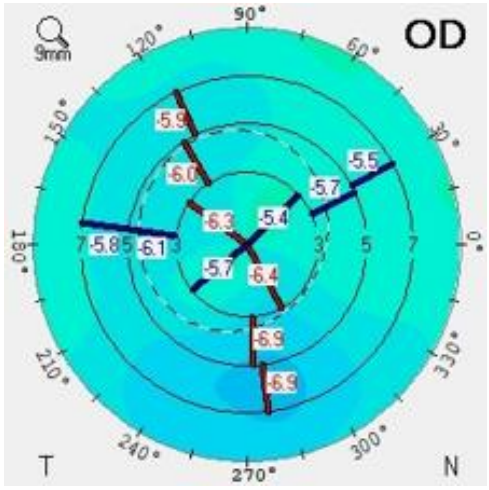


**Figure 7.** Mapa topográfico corneal Sag_P.

Independientemente de que fuente consulte para obtener los datos, es decir, el alojamiento en GitHub del presente proyecto o la carpeta de Google Drive de la publicación original, usted obtendrá la siguiente estructura de almacenamiento:

- Carpeta 1: Conjunto de entrenamiento.
  - Sub carpeta 1: No queratocono
    150 casos almacenados en 150 carpetas, cada carpeta con 7 mapas corneales
  - Sub carpeta 2: queratocono.
    150 casos almacenados en 150 carpetas, cada carpeta con 7 mapas corneales
  - Sub carpeta 3: Sospechoso de queratocono.
    123 casos almacenados en 123 carpetas, cada carpeta con 7 mapas corneales
- Carpeta 2: Conjunto de Prueba.
  - Sub carpeta 1: No queratocono
    50 casos almacenados en 50 carpetas, cada carpeta con 7 mapas corneales
  - Sub carpeta 2: queratocono.
    50 casos almacenados en 50 carpetas, cada carpeta con 7 mapas corneales
  - Sub carpeta 3: Sospechoso de queratocono.
    50 casos almacenados en 50 carpetas, cada carpeta con 7 mapas corneales

La obtención de los datos dentro del sistema se realizo de la siguiente manera.

- Obtención de los datos.
  Se generaron 2 archivos de texto plano tipo CSV, bajo los nombres ‘train_validation_path.csv’ e ‘independent_test_path.csv’, en los cuales se almacenan las direcciones path relativas de los 7 mapas topográficos corneales por caso, así como una etiqueta numérica al caso, que indica si el paciente tiene queratocono, el paciente es sospechoso de queratocono o es un paciente con ojos normales.
- Carga de los datos.
  Los datos son cargados al “Sistema de diagnóstico de queratocono basado en inferencia bayesiana”, por medio de los archivos ‘train_validation_path.csv’ e ‘independent_test_path.csv’, los cuales son leídos para obtener las rutas relativas de los mapas topográficos corneales, almacenados en formato JPG de 224*224*3.

### 3.2. Preprocesamiento de Datos.

El preprocesamiento de los datos dentro del sistema se realizará de la siguiente manera.

- Proceso de limpieza: Se aplicó a todos los mapas topográficos corneales un proceso de limpieza para la eliminación de ruido en las imágenes, ello con el fin de no interferir en el modelo.
- Normalización: Se aplicó normalización a todos los mapas topográficos, dividiendo los valores de píxeles por 255, para ajustarlos al rango [0, 1] y asegurar una buena convergencia del modelo.

### 3.3. Transformación y Preparación de Dato.

- Aplanamiento de Imágenes: Se aplicó aplanado a los mapas topográficos corneales, para transformar las imágenes en vectores unidimensionales, técnica necesaria para el proyecto a realizar.
- Estandarización: Se aplicó escalamiento estándar para asegurar obtener características con media de 0 y desviación estándar de 1, con el fin de evitar que características más grandes interfieran y dominen el modelo.

### 3.4. Reducción de Dimensionalidad.

Se utilizó Análisis de Componentes Principales (PCA) para reducir la dimensionalidad de los datos. Se evaluó el rendimiento del modelo utilizando diferentes números de componentes principales.

### 3.5. Modelado y Evaluación.

- Modelo Naive Bayes: Se desarrollo y entreno un clasificador por medio de Naive Bayes Gaussiano, utilizando los datos reducidos por PCA. Naive Bayes es un clasificador probabilístico simple pero efectivo que se basa en el teorema de Bayes y asume independencia entre las características.
- Evaluación del Modelo: Se evalúa el rendimiento del modelo en el conjunto de entrenamiento y prueba mediante la precisión y los informes de clasificación.

### 3.6. Resultados.

- Análisis de Varianza: Se grafica la varianza explicada por cada número de componentes principales.
- Precisión del Modelo: Se grafica la precisión del modelo en el conjunto de entrenamiento y prueba en función del número de componentes principales.

## 4. Marco Teórico.

### 4.1. PCA (Análisis de Componentes Principales):

El análisis de componentes principales consiste en analizar un conjunto de datos de entrada, el cual contiene diferentes observaciones descritas por múltiples variables independientes o dependientes y cuyas relaciones entre sí no tienen por qué conocerse. Su objetivo principal es reducir la dimensión del conjunto de datos de entrada, pero manteniendo la mayor cantidad de información posible para poder analizarlos de forma más fácil poder simplificar los criterios de decisión. El análisis de componentes principales se ha convertido en una técnica popular debido a dos propiedades:

- Garantiza la mínima pérdida de información dado que los componentes principales obtienen secuencialmente la máxima variabilidad.
- Las componentes principales obtenidos pueden tratarse independientemente y su procesado es más fácil dado que son ortogonales entre sí.

La labor de PCA es hallar una matriz que transforme linealmente el espacio de los datos de entrada de la matriz $X$ en una nueva matriz $Y$ con un menor número de características.

### 4.2. Naive Bayes Gaussiano.

El clasificador Naive Bayes Gaussiano es una variante del clasificador Naive Bayes, el cual es un algoritmo de aprendizaje automático supervisado utilizado para tareas de clasificación (¿Qué Son los Clasificadores Naive Bayes? | IBM, s. f.).

Naive Bayes Gaussiano, basado en teorema de Bayes, es un clasificador ingenuo, es decir, el clasificador asume que las características son continuas, tienen una distribución normal y son independientes entre sí.

El teorema de Bayes describe la probabilidad de un evento con base en el conocimiento previo de condiciones y su relación. Matemáticamente, es de la forma:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{1}$$

donde:

- $P(A|B)$: probabilidad de A dado B (probabilidad a posteriori de la clase).
- $P(B|A)$: probabilidad de B dado A (probabilidad de ver la evidencia dada la hipótesis, verosimilitud (likelihood)).
- $P(A)$: probabilidad de A (probabilidad previa de la hipótesis, probabilidad a priori de la clase).
- $P(B)$: probabilidad de B (probabilidad de la evidencia, verosimilitud marginal, probabilidad a priori del vector de atributos)).

La probabilidad $P(B|A)$ es:

$$P(B|A) = P(B_1|A)P(B_2|A)\cdots P(B_n|A) = \prod_{i=1}^{n} P(B_i|A) \tag{2}$$

En Naive-Bayes, estimar la verosimilitud $P(B_i|A)$ dependerá de los atributos. Si los atributos son discretos se cuentan las frecuencias. Si los atributos son continuos, se distribuye cada atributo de acuerdo a una normal. Y se aplica la función de densidad gaussiana.

$$P(x_i|c) = \phi(x_i; \mu_i^c, (\sigma_i^c)^2) = \frac{1}{\sqrt{2\pi}\sigma_i^c} exp\left(-\frac{(x_i - \mu_i^c)^2}{2(\sigma_i^c)^2}\right) \tag{3}$$

#### *4.2.1. Características de la clasificación Naïve-Bayes.*

Ventajas:

- Simple, rápido y efectivo.
- Da buenos resultados con datos con ruido.
- Requiere relativamente pocos registros para entrenar, y también funciona bien con grandes conjuntos de datos.
- Proporciona fácilmente la probabilidad estimada para cada predicción.

Inconvenientes:

- Se sustenta de una falsa independencia entre tributos.
- erdida de rendimiento con demasiados atributos numéricos.
- La clasificación predicha 'A' es más confiable que la probabilidad estimada $P(A|B)$.

## 5. Propuesta.

Se elaboraron dos notebooks de Python para este proyecto. El primero a utilizarse, de nombre '10056926-BSeIB-Recurso.ipynb' se generó con la finalidad de poder obtener las rutas path relativas de los mapas topográficos corneales. El segundo, de nombre '10056926-BustamanteJ-BSeIB-Proy.ipynb' es donde se generó y documento el proyecto "Sistema de diagnóstico de queratocono basado en inferencia bayesiana".
A continuación se describen las actividades realizadas en dichos documentos.

### 5.1. 10056926-BSeIB-Recurso

Importación de módulos, necesario para manejar archivos del sistema, procesar datos estructurados en CSV, realizar operaciones avanzadas de manipulación de texto, y trabajar con imágenes y datos numéricos.

```python
import os
import csv
import re
from PIL import Image #type: ignore
import numpy as np    #type: ignore
```

**Code 1.** Módulos.

Definición y configuración de las rutas para los directorios principales del proyecto. Si usted está replicando este proyecto, considere sustituir las direcciones.

```python
base_dir = r'C:\Users\Busta\Documents\GitHub\BSeIB_Keratoconus'
train_validation_dir = os.path.join(base_dir, 'Train_Validation sets')
independent_test_dir = os.path.join(base_dir, 'Independent Test Set')
```

**Code 2.** Direcciones.

Definición y configuración de prefijos y etiquetas.

```python
label_map = {
    0: 'Normal',
    1: 'Suspect',
    2: 'Keratoconus'
}
```

**Code 3.** Mapeo de etiquetas.

```python
prefix_map = {
    0: 'NOR',
    1: 'SUSP',
    2: 'KCN'
}
```

**Code 4.** Prefijos según la etiqueta.

```python
def get_image_paths(root_folder, label_map, prefix_map):
    data = []
    patient_id = 1

    for label, label_name in label_map.items():
        label_folder = os.path.join(root_folder, label_name)
        cases = os.listdir(label_folder)

        for case in cases:
            case_folder = os.path.join(label_folder, case)
            image_files = os.listdir(case_folder)

            # Filtrar solo archivos de im genes
            image_files = [file for file in image_files if file.endswith('.jpg')]

            if len(image_files) != 7:
                continue

            prefix = prefix_map[label]
            case_number = case.split('case')[-1]

            CT_A = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_CT_A.jpg'), base_dir)
            EC_A = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_EC_A.jpg'), base_dir)
            EC_P = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_EC_P.jpg'), base_dir)
            Elv_A = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_Elv_A.jpg'), base_dir)
            Elv_P = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_Elv_P.jpg'), base_dir)
            Sag_A = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_Sag_A.jpg'), base_dir)
            Sag_P = os.path.relpath(os.path.join(case_folder, f'{prefix}_{case_number}_Sag_P.jpg'), base_dir)

            data.append({
                'PatientID': patient_id,
                'Label': label,
                'CT_A': CT_A,
                'EC_A': EC_A,
                'EC_P': EC_P,
                'Elv_A': Elv_A,
                'Elv_P': Elv_P,
                'Sag_A': Sag_A,
                'Sag_P': Sag_P
            })
            patient_id += 1

    return data
```

**Code 5.** Función obtener direcciones.

Escribir archivo CSV con los datos obtenidos para 'Train_Validation sets' e 'Independent Test Set'.

```python
# Para el entrenamiento.

train_data = get_image_paths(
    train_validation_dir, label_map, prefix_map)

train_csv_file = os.path.join(base_dir, 'train_validation_path.csv')
csv_columns = ['PatientID', 'Label', 'CT_A', 'EC_A', 'EC_P', 'Elv_A', 'Elv_P', 'Sag_A', 'Sag_P']

with open(train_csv_file, 'w', newline='') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=csv_columns)
    writer.writeheader()
    for data in train_data:
        writer.writerow(data)

print(f'Archivo CSV para Train_Validation sets generado: {train_csv_file}')

# Para validar.

test_data = get_image_paths(independent_test_dir, label_map, prefix_map)

test_csv_file = os.path.join(base_dir, 'independent_test_path.csv')

with open(test_csv_file, 'w', newline='') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=csv_columns)
    writer.writeheader()
    for data in test_data:
        writer.writerow(data)

print(f'Archivo CSV para Independent Test Set generado: {test_csv_file}')
```

**Code 6.** Obtener archivos CSV.

### 5.2. 10056926-BustamanteJ-BSeIB-Proy

```python
import warnings
```

```
warnings.filterwarnings('ignore')

import pandas as pd
from skimage.io import imread
from skimage.transform import resize
from skimage.color import rgb2gray
from skimage.filters import threshold_otsu
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import
    classification_report, accuracy_score,
    confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns


train_validation_path = pd.read_csv('
    train_validation_path.csv')
independent_test_path = pd.read_csv('
    independent_test_path.csv')


def remove_text_symbols(image):
    gray_image = rgb2gray(image)
    threshold = threshold_otsu(gray_image)
    binary_image = gray_image > threshold
    return binary_image


def load_and_preprocess_image(path):
    image = imread(path)
    resized_image = resize(image, (224, 224))
    cleaned_image = remove_text_symbols(
    resized_image)
    return cleaned_image


def load_images(df):
    images = []
    labels = []
    for index, row in df.iterrows():
        patient_images = []
        for col in ['CT_A', 'EC_A', 'EC_P', '
    Elv_A', 'Elv_P', 'Sag_A', 'Sag_P']:
            image_path = row[col]
            patient_images.append(
    load_and_preprocess_image(image_path))
        images.append(patient_images)
        labels.append(row['Label'])
    return images, labels


def normalize_images(images):
    normalized_images = []
    for image_set in images:
        normalized_set = [image / 255.0 for
    image in image_set]
        normalized_images.append(normalized_set)
    return normalized_images


def flatten_images(images):
    flattened_images = []
    for image_set in images:
        flattened_set = [image.flatten() for
    image in image_set]
        flattened_images.append(np.concatenate(
    flattened_set))
    return np.array(flattened_images)


def standardScaler(xTrain, xTest):
    scaler = StandardScaler()
    xTrain_scaled = scaler.fit_transform(xTrain)
    xTest_scaled = scaler.transform(xTest)
    return xTrain_scaled, xTest_scaled


def reduce_dimension(images, n_components):
    pca = PCA(n_components=n_components)
    reduced_images = pca.fit_transform(images)
    return reduced_images, pca


def train_and_evaluate_bayes(train_data,
    train_labels, test_data, test_labels):
    model = GaussianNB()
    model.fit(train_data, train_labels)
    train_predictions = model.predict(train_data
    )
    test_predictions = model.predict(test_data)
    train_accuracy = accuracy_score(train_labels
    , train_predictions)
    test_accuracy = accuracy_score(test_labels,
    test_predictions)
    train_report = classification_report(
    train_labels, train_predictions)
    test_report = classification_report(
    test_labels, test_predictions)
    train_conf_matrix = confusion_matrix(
    train_labels, train_predictions)
    test_conf_matrix = confusion_matrix(
    test_labels, test_predictions)
    return train_accuracy, test_accuracy,
    train_report, test_report, train_conf_matrix
    , test_conf_matrix


def display_preprocessed_images(images,
    num_samples=5):
    fig, axes = plt.subplots(num_samples, 7,
    figsize=(15, 15))
    for i in range(num_samples):
        for j in range(7):
            axes[i, j].imshow(images[i][j], cmap
    ='gray')
            axes[i, j].axis('off')
    plt.show()


train_images, train_labels = load_images(
    train_validation_path)
test_images, test_labels = load_images(
    independent_test_path)


train_images_normalized = normalize_images(
    train_images)
test_images_normalized = normalize_images(
    test_images)


display_preprocessed_images(
    train_images_normalized)


train_images_flattened = flatten_images(
    train_images_normalized)
test_images_flattened = flatten_images(
    test_images_normalized)


xTrain_scaled, xTest_scaled = standardScaler(
    train_images_flattened,
    test_images_flattened)


n_components_range = range(1, 151, 15)


train_accuracies = []
test_accuracies = []
variances = []
train_conf_matrices = []
test_conf_matrices = []


for n_components in n_components_range:

```

```
    reduced_train_images, pca = reduce_dimension(xTrain_scaled, n_components)


    reduced_test_images = pca.transform(xTest_scaled)


    train_accuracy, test_accuracy, train_report, test_report, train_conf_matrix, test_conf_matrix = train_and_evaluate_bayes(reduced_train_images, train_labels, reduced_test_images, test_labels)


    train_accuracies.append(train_accuracy)
    test_accuracies.append(test_accuracy)
    variances.append(np.sum(pca.explained_variance_ratio_))
    train_conf_matrices.append(train_conf_matrix)
    test_conf_matrices.append(test_conf_matrix)

    print(f"Components: {n_components}, Train Accuracy: {train_accuracy:.4f}, Test Accuracy: {test_accuracy:.4f}, Variance: {np.sum(pca.explained_variance_ratio_):.4f}")
    print("Train Confusion Matrix:\n", train_conf_matrix)
    print("Test Confusion Matrix:\n", test_conf_matrix)
    print("Train Classification Report:\n", train_report)
    print("Test Classification Report:\n", test_report)


plt.figure(figsize=(18, 12))

plt.subplot(2, 2, 1)
plt.plot(n_components_range, variances, marker='o')
plt.xlabel('Number of Components')
plt.ylabel('Explained Variance')
plt.title('Explained Variance vs Number of Components')

plt.subplot(2, 2, 2)
plt.plot(n_components_range, train_accuracies, label='Train Accuracy', marker='o')
plt.plot(n_components_range, test_accuracies, label='Test Accuracy', marker='o')
plt.xlabel('Number of Components')
plt.ylabel('Accuracy')
plt.title('Accuracy vs Number of Components')
plt.legend()


fig, axes = plt.subplots(len(n_components_range), 2, figsize=(20, 40))
fig.suptitle('Confusion Matrices for Different Number of PCA Components', fontsize=20)
for i, (train_conf_matrix, test_conf_matrix) in enumerate(zip(train_conf_matrices, test_conf_matrices)):
    sns.heatmap(train_conf_matrix, annot=True, fmt='d', cmap='Blues', ax=axes[i, 0])
    axes[i, 0].set_title(f'Train Confusion Matrix (Components: {n_components_range[i]})')
    axes[i, 0].set_xlabel('Predicted Label')
    axes[i, 0].set_ylabel('True Label')

    sns.heatmap(test_conf_matrix, annot=True, fmt='d', cmap='Blues', ax=axes[i, 1])
    axes[i, 1].set_title(f'Test Confusion Matrix (Components: {n_components_range[i]})')
    axes[i, 1].set_xlabel('Predicted Label')
    axes[i, 1].set_ylabel('True Label')

plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```

**Code 7.** Proyecto v1

## 6. Resultados.

### 6.1. 10056926-BSeIB-Recurso

Dos archivos CSV que almacenan las direcciones relativas de los 7 mapas topográficos corneales por paciente.

```
PatientID,Label,CT_A,EC_A,EC_P,Elv_A,Elv_P,Sag_A,Sag_P
1,0,Independent Test Set\Normal\case1\NOR_1_CT_A.jpg,Independent Test Set\Normal\case1\NOR_1_EC_A.jpg,Independent Test Set\Normal\case1\NOR_1_EC_P.jpg,Independent Test Set\Normal\case1\NOR_1_Elv_A.jpg,Independent Test Set\Normal\case1\NOR_1_Elv_P.jpg,Independent Test Set\Normal\case1\NOR_1_Sag_A.jpg,Independent Test Set\Normal\case1\NOR_1_Sag_P.jpg
2,0,Independent Test Set\Normal\case10\NOR_10_CT_A.jpg,Independent Test Set\Normal\case10\NOR_10_EC_A.jpg,Independent Test Set\Normal\case10\NOR_10_EC_P.jpg,Independent Test Set\Normal\case10\NOR_10_Elv_A.jpg,Independent Test Set\Normal\case10\NOR_10_Elv_P.jpg,Independent Test Set\Normal\case10\NOR_10_Sag_A.jpg,Independent Test Set\Normal\case10\NOR_10_Sag_P.jpg
3,0,Independent Test Set\Normal\case11\NOR_11_CT_A.jpg,Independent Test Set\Normal\case11\NOR_11_EC_A.jpg,Independent Test Set\Normal\case11\NOR_11_EC_P.jpg,Independent Test Set\Normal\case11\NOR_11_Elv_A.jpg,Independent Test Set\Normal\case11\NOR_11_Elv_P.jpg,Independent Test Set\Normal\case11\NOR_11_Sag_A.jpg,Independent Test Set\Normal\case11\NOR_11_Sag_P.jpg
4,0,Independent Test Set\Normal\case12\NOR_12_CT_A.jpg,Independent Test Set\Normal\case12\NOR_12_EC_A.jpg,Independent Test Set\Normal\case12\NOR_12_EC_P.jpg,Independent Test Set\Normal\case12\NOR_12_Elv_A.jpg,Independent Test Set\Normal\case12\NOR_12_Elv_P.jpg,Independent Test Set\Normal\case12\NOR_12_Sag_A.jpg,Independent Test Set\Normal\case12\NOR_12_Sag_P.jpg
5,0,Independent Test Set\Normal\case13\NOR_13_CT_A.jpg,Independent Test Set\Normal\case13\NOR_13_EC_A.jpg,Independent Test Set\Normal\case13\NOR_13_EC_P.jpg,Independent Test Set\Normal\case13\NOR_13_Elv_A.jpg,Independent Test Set\Normal\case13\NOR_13_Elv_P.jpg,Independent Test Set\Normal\case13\NOR_13_Sag_A.jpg,Independent Test Set\Normal\case13\NOR_13_Sag_P.jpg
6,0,Independent Test Set\Normal\case14\NOR_14_CT_A.jpg,Independent Test Set\Normal\case14\NOR_14_EC_A.jpg,Independent Test Set\Normal\case14\NOR_14_EC_P.jpg,Independent Test Set\Normal\case14\NOR_14_Elv_A.jpg,Independent Test Set\Normal\case14\NOR_14_Elv_P.jpg,Independent Test Set\Normal\case14\NOR_14_Sag_A.jpg,Independent Test Set\Normal\case14\NOR_14_Sag_P.jpg
7,0,Independent Test Set\Normal\case15\NOR_15_CT_A.jpg,Independent Test Set\Normal\case15\NOR_15_EC_A.jpg,Independent Test Set\Normal\case15\NOR_15_EC_P.jpg,Independent Test Set\Normal\case15\NOR_15_Elv_A.jpg,Independent Test Set\Normal\case15\NOR_15_Elv_P.jpg,Independent Test Set\Normal\case15\NOR_15_Sag_A.jpg,Independent Test Set\Normal\case15\NOR_15_Sag_P.jpg
```

**Code 8.** independent test CSV.

```
PatientID,Label,CT_A,EC_A,EC_P,Elv_A,Elv_P,Sag_A,Sag_P
1,0,Train_Validation sets\Normal\case1\NOR_1_CT_A.jpg,Train_Validation sets\Normal\case1\NOR_1_EC_A.jpg,Train_Validation sets\Normal\case1\NOR_1_EC_P.jpg,Train_Validation sets\Normal\case1\NOR_1_Elv_A.jpg,Train_Validation sets\Normal\case1\NOR_1_Elv_P.jpg,Train_Validation sets\Normal\case1\NOR_1_Sag_A.jpg,Train_Validation sets\Normal\case1\NOR_1_Sag_P.jpg
2,0,Train_Validation sets\Normal\case10\NOR_10_CT_A.jpg,Train_Validation sets\Normal\case10\NOR_10_EC_A.jpg,Train_Validation sets\Normal\case10\NOR_10_EC_P.jpg,Train_Validation sets\Normal\case10\NOR_10_Elv_A.jpg,Train_Validation sets\Normal\case10\NOR_10_Elv_P.jpg,Train_Validation sets\Normal\case10\NOR_10_Sag_A.jpg,Train_Validation sets\Normal\case10\NOR_10_Sag_P.jpg
3,0,Train_Validation sets\Normal\case100\NOR_100_CT_A.jpg,Train_Validation sets\Normal\case100\NOR_100_EC_A.jpg,Train_Validation sets\Normal\case100\NOR_100_EC_P.jpg,Train_Validation sets\Normal\case100\NOR_100_Elv_A.jpg,Train_Validation sets\Normal\case100\NOR_100_Elv_P.jpg,Train_Validation sets\Normal\case100\NOR_100_Sag_A.jpg,Train_Validation sets\Normal\case100\NOR_100_Sag_P.jpg
4,0,Train_Validation sets\Normal\case101\NOR_101_CT_A.jpg,Train_Validation sets\Normal\case101\NOR_101_EC_A.jpg,Train_Validation sets\Normal\case101\NOR_101_EC_P.jpg,Train_Validation sets\Normal\case101\NOR_101_Elv_A.jpg,Train_Validation sets\Normal\case101\NOR_101_Elv_P.jpg,Train_Validation sets\Normal\case101\NOR_101_Sag_A.jpg,Train_Validation sets\Normal\case101\NOR_101_Sag_P.jpg
5,0,Train_Validation sets\Normal\case102\NOR_102_CT_A.jpg,Train_Validation sets\Normal\case102\NOR_102_EC_A.jpg,Train_Validation sets\Normal\case102\NOR_102_EC_P.jpg,Train_Validation sets\Normal\case102\NOR_102_Elv_A.jpg,Train_Validation sets\Normal\case102\NOR_102_Elv_P.jpg,Train_Validation sets\Normal\case102\NOR_102_Sag_A.jpg,Train_Validation sets\Normal\case102\NOR_102_Sag_P.jpg
6,0,Train_Validation sets\Normal\case103\NOR_103_CT_A.jpg,Train_Validation sets\Normal\case103\NOR_103_EC_A.jpg,Train_Validation sets\Normal\case103\NOR_103_EC_P.jpg,Train_Validation sets\Normal\case103\NOR_103_Elv_A.jpg,Train_Validation sets\Normal\case103\NOR_103_Elv_P.jpg,Train_Validation sets\Normal\case103\NOR_103_Sag_A.jpg,Train_Validation sets\Normal\case103\NOR_103_Sag_P.jpg
7,0,Train_Validation sets\Normal\case104\NOR_104_CT_A.jpg,Train_Validation sets\Normal\case104\NOR_104_EC_A.jpg,Train_Validation sets\Normal\case104\NOR_104_EC_P.jpg,Train_Validation sets\Normal\case104\NOR_104_Elv_A.jpg,Train_Validation sets\Normal\case104\NOR_104_Elv_P.jpg,Train_Validation sets\Normal\case104\NOR_104_Sag_A.jpg,Train_Validation sets\Normal\case104\NOR_104_Sag_P.jpg
```

**Code 9.** train validation CSV.

### 6.2. 10056926-BustamanteJ-BSeIB-Proy

Implementación del método de Otsu para la binarización de imágenes. Esta técnica de umbralización global, calcula un valor de umbral óptimo para separar los píxeles de una imagen en dos clases: fondo y objeto (o primer plano y fondo). Su función es maximizar la varianza interclase y minimizar la varianza intraclase, de modo que el valor de umbral seleccionado sea el que mejor separe las dos clases en términos de contraste.

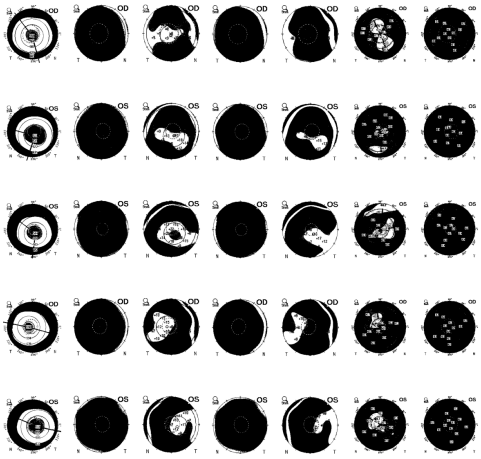

**Figure 8.** Mapas topográficos corneales - threshold$_o tsu$.

```
Components: 1, Train Accuracy: 0.5981, Test Accuracy: 0.5467, Variance: 0.1164
Train Confusion Matrix:
 [[115   3  32]
 [ 90   2  31]
 [ 13   1 136]]
Test Confusion Matrix:
 [[38  1 11]
 [25  3 22]
```

**Code 10.** Reportes de la clasificacion.

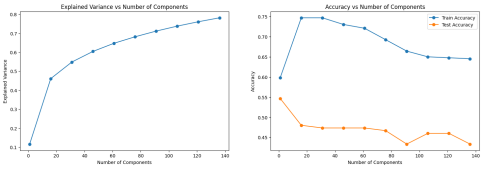


**Figure 9.** Comparación de Accuracy.

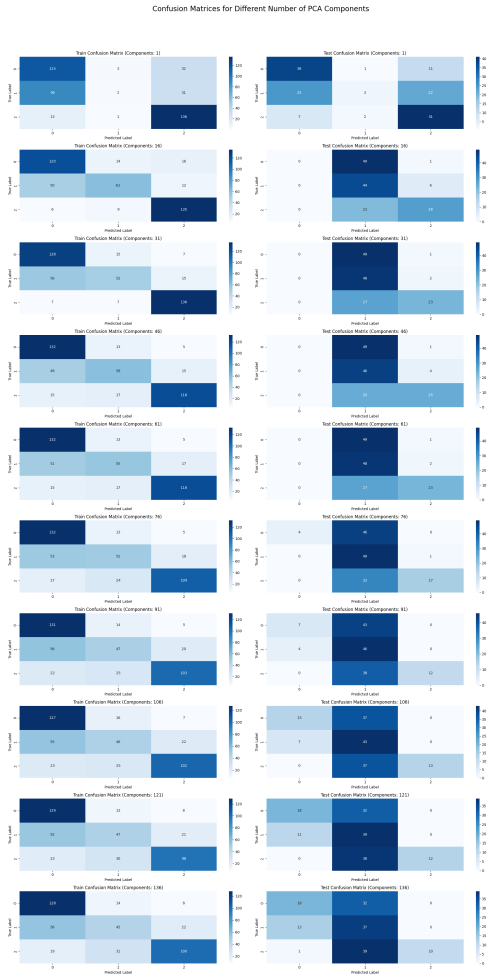


**Figure 10.** Matrices de confusion.

## 7. Conclusiones.

Es posible observar que a medida que aumentan los componentes principales, el modelo tiende a sobre ajustarse al conjunto de entrenamiento, lo que resulta en una disminución del accuracy en el conjunto de prueba. Aunado a ello, la clase 0 (sin queratocono) presenta errores significativos en varios casos, y la clase 1 (sospechoso de queratocono) presenta un bajo f1-score en la mayoría de los casos, lo que indica dificultad para detectar esta categoría. Dada la naturaleza del Naive Bayes, es posible suponer, que se obtendrán mejoras considerables evaluando modelos de clasificación más avanzados, o en su caso, considerar otros métodos de regularización y reducción de dimensionalidad de características, con el fin de mitigar el sobre-ajuste.

| Componentes | Train Acc. | Test Acc. | Varianza | Análisis |
|---|---|---|---|---|
| 1 | 0.5981 | 0.5467 | 0.1164 | Bajo accuracy, f1-score bajo para clase 1. |
| 16 | 0.7447 | 0.4800 | 0.4611 | Mejora en train, sobreajuste en test, f1-score bajo para clase 1. |
| 31 | 0.7494 | 0.4667 | 0.5485 | Capacidad mejorada para clase 1 en test, sobreajuste persistente. |
| 46 | 0.7281 | 0.4533 | 0.6043 | Sobreajuste, f1-score bajo para clase 1. |
| 61 | 0.7163 | 0.4400 | 0.6472 | Disminución en accuracy, aumento en error para clase 0 en test. |
| 76 | 0.6809 | 0.4667 | 0.6818 | Bajo rendimiento general en test, errores en clasificación clase 0. |
| 91 | 0.6714 | 0.4600 | 0.7115 | Aumento de error para clase 0 en test. |
| 106 | 0.6596 | 0.4533 | 0.7375 | F1-score clase 0 en test disminuye aún más. |
| 121 | 0.6525 | 0.4533 | 0.7606 | Persiste bajo rendimiento para clase 0 en test. |
| 136 | 0.6383 | 0.4400 | 0.7816 | F1-score clase 0 en test muy bajo. |

**Table 1.** Análisis de los resultados del PCA

## 8. Contacto.

Para obtener más información, resolver dudas o enviar sugerencias, no dude en ponerse en contacto conmigo.

https://github.com/BustamanteJ2/BSeIB$_K$*eratoconus*
eduardo.bustamantej@uaem.edu.mx

## Referencias


- ¿Qué son los clasificadores NaÏve Bayes? | IBM. (s. f.). Recuperado 11 de junio de 2024, de https://www.ibm.com/mx-es/topics/naive-bayes
- 02.1 Clasificación naive-Bayes — Introducción al Aprendizaje Automático. (s. f.). Recuperado 11 de junio de 2024, de https://dcain.etsin.upm.es/ carlos/bookAA/02.1_Metodos deClasificacion-Naive-Bayes.html
- Diaz, B., Diaz, B. (2022, 21 junio). 7 Estadística y modelos naive, el campo base de la IA. ImpulsateK - Artificial intelligence, tools, insights and wisdom gleaned from the knowledge of others. Recuperado 11 de junio de 2024, de https://impulsatek.com/7-estadistica-y-modelos-naive-el-campo-base-de-la-ia/
- Jauregui, A. F. (2023, 19 agosto). Clasificación de Texto con Naive Bayes en Python. Ander Fernández. Recuperado 11 de junio de 2024, de https://anderfernandez.com/blog/naive-bayes-en-python/
- Ibarra, J. M. (2023, 19 marzo). Machine Learning 101 — Clasificador Naive Bayes - Juan Manuel Ibarra - medium. Medium. https://jmibarra86.medium.com/machine-learning-101-clasificador-naive-bayes-36b67fe5b6a9
- H. Zou, T. Hastie y R. Tibshirani, "Sparse Principal Component Analysis". Journal of Computational and Graphical Statistics, 15, pp.265-286. 2006.
- Team, L. D. (2021, 3 marzo). How is Keratoconus treated. NKCF.org. https://nkcf.org/how-is-keratoconus-treated/
- Especialistas del IMSS ofrecen tratamiento para el Queratocono. (2019, abril). http://www.imss.gob.mx/prensa/archivo/201904/075
- Cornea Research Foundation of America - Keratoconus. (s. f.). https://www.cornea.org/learning-center/conditions-research-areas/keratoconus.aspx
- Al-Timemy, A. H., Mosa, Z. M., Alyasseri, Z., Lavric, A., Lui, M. M., Hazarbassanov, R. M., Yousefi, S. (2021). A Hybrid Deep Learning Construct for Detecting Keratoconus From Corneal Maps. Translational vision science technology, 10(14), 16. https://doi.org/10.1167/tvst.10.14.16