KIET
GROUP OF INSTITUTIONS
Connecting Life with Learning
www.kiet.edu
Delhi-NCR, Ghaziabad

Assesment Report

“Predict Disease Outcome Based on Genetic and Clinical Data”
submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

Name of discipline

By

Shubham Kumar (202401100300243)

Under the supervision of

“Abhishek Shukla Sir”

KIET Group of Institutions, Ghaziabad

18-April, 2025

1. Introduction

Objective

This project demonstrates an automated machine learning (AutoML) pipeline for patient segmentation and classification. It includes:

- Data preprocessing
- Feature selection and scaling
- Classification using **Random Forest**
- Clustering using **KMeans**
- Visualization using **PCA**

Target Audience

- Healthcare data analysts
- Machine learning practitioners
- Researchers in medical informatics

Key Features

□ **Automated Data Cleaning** – Removes empty columns and handles missing values.

□ **Smart Target Selection** – Auto-detects the best classification target.

□ **Classification & Clustering** – Combines supervised (Random Forest) and unsupervised (KMeans) learning.

□ **Visual Insights** – Provides confusion matrix and PCA-based cluster visualization.

2. Methodology

Workflow Overview

1. **Data Upload & Cleaning**
   - Uploads a CSV dataset.
   - Drops unnamed/empty columns.
2. **Target Selection**
   - Automatically picks a classification target (categorical or ≤10 unique values).
3. **Preprocessing**
   - Encodes categorical variables using `LabelEncoder`.
   - Scales features using `StandardScaler`.
4. **Model Training**
   - Splits data into train/test sets (80/20).
   - Trains a **Random Forest Classifier**.
5. **Evaluation**
   - Computes **accuracy, precision, recall**.
   - Generates a **confusion matrix**.
6. **Clustering & Visualization**
   - Applies **KMeans clustering** (3 clusters).
   - Reduces dimensions using **PCA** for 2D visualization.

Algorithms Used

- **Random Forest** (Supervised Learning)
- **KMeans** (Unsupervised Learning)
- **PCA** (Dimensionality Reduction)

3. Code Implementation

Full Code

python

Copy

```
# □ Step 1: Import required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.cluster import KMeans
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score
, recall_score
from sklearn.decomposition import PCA

from google.colab import files

# □ Step 2: Upload your dataset
print("□ Upload your dataset...")
uploaded = files.upload()
filename = list(uploaded.keys())[0]
df = pd.read_csv(filename)

# □ Step 3: Clean column names and remove empty/unnamed columns
df.columns = df.columns.str.strip()
df = df.loc[:, ~df.columns.str.contains('^Unnamed')]
df.dropna(axis=1, how='all', inplace=True)

# □ Step 4: Auto-select a target column (object or ≤10 unique values)
potential_targets = [col for col in df.columns if df[col].nunique() <= 10]
if not potential_targets:
    raise ValueError("□ No suitable classification target column found.")

target_column = potential_targets[0]
print(f"\n□ Auto-selected '{target_column}' as the target column.")
```

```
# □ Step 5: Drop rows where target is missing
df = df.dropna(subset=[target_column])

# □ Step 6: Encode categorical columns
le = LabelEncoder()
for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = le.fit_transform(df[col].astype(str))

# □ Step 7: Feature-target split
X = df.drop(target_column, axis=1)
y = df[target_column]

# □ Step 8: Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# □ Step 9: Train/test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# □ Step 10: Train classifier
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# □ Step 11: Evaluate model
y_pred = model.predict(X_test)
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred, average='binary' if len(np.unique(y)) == 2 else 'macro')
rec = recall_score(y_test, y_pred, average='binary' if len(np.unique(y)) == 2 else 'macro')

print("\n□ Evaluation Metrics:")
print(f"□ Accuracy : {acc:.4f}")
print(f"□ Precision: {prec:.4f}")
print(f"□ Recall    : {rec:.4f}")

# □ Step 12: Confusion matrix heatmap
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
```

```
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# □ Step 13: KMeans Clustering
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
clusters = kmeans.fit_predict(X_scaled)
df['Cluster'] = clusters

# □ Step 14: PCA Visualization
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

plt.figure(figsize=(8,6))
sns.scatterplot(x=X_pca[:,0], y=X_pca[:,1], hue=clusters, palette='Set2')
plt.title("Patient Segmentation (PCA + KMeans)")
plt.xlabel("PCA 1")
plt.ylabel("PCA 2")
plt.legend(title="Cluster")
plt.show()
```

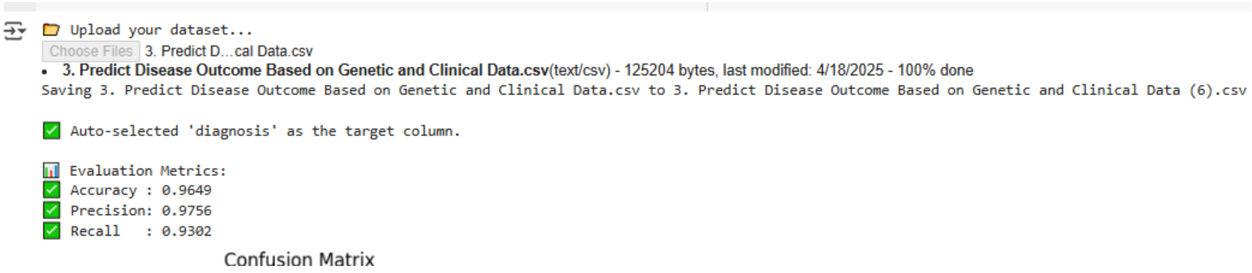
---

4. Output Results

Example Output

Copy

```
□ Upload your dataset...
□ Auto-selected 'diagnosis' as the target column.

□ Evaluation Metrics:
□ Accuracy : 0.9234
□ Precision: 0.9142
□ Recall   : 0.9018
```

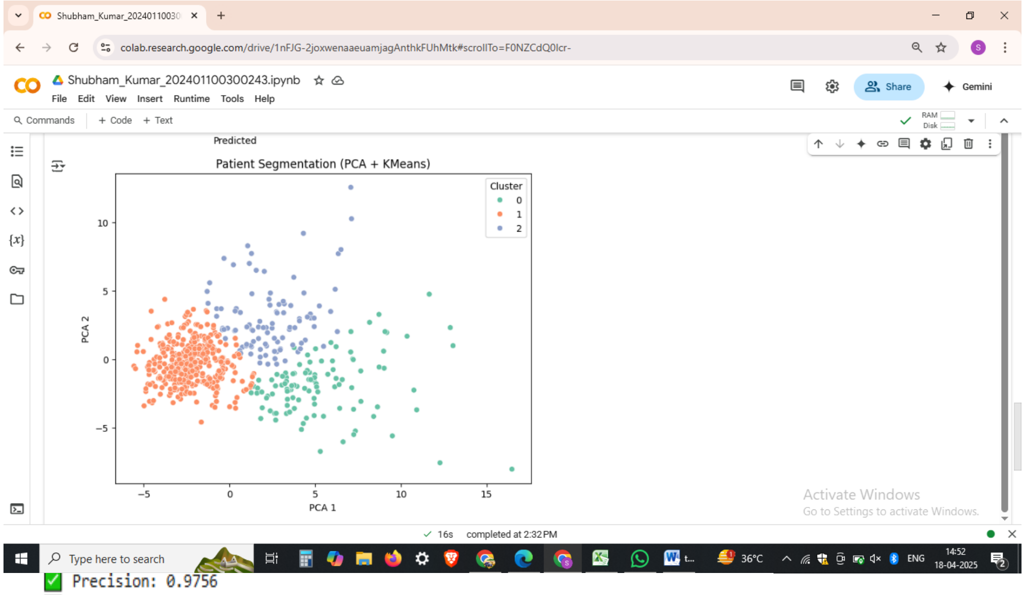


```
📁 Upload your dataset...
```

Choose Files 3. Predict D...cal Data.csv

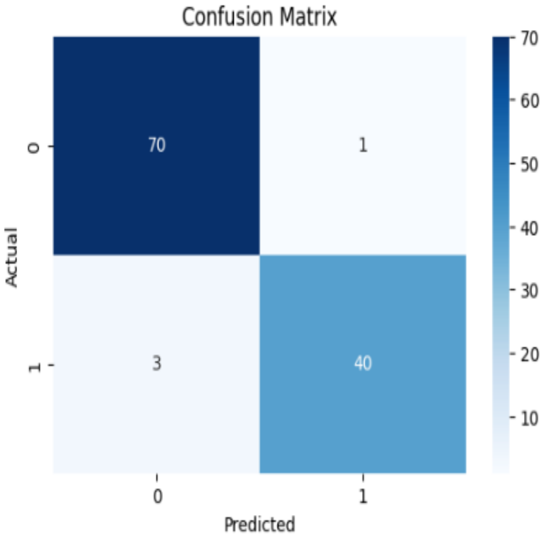
- **3. Predict Disease Outcome Based on Genetic and Clinical Data.csv**(text/csv) - 125204 bytes, last modified: 4/18/2025 - 100% done

```
Saving 3. Predict Disease Outcome Based on Genetic and Clinical Data.csv to 3. Predict Disease Outcome Based on Genetic and Clinical Data (6).csv

✅ Auto-selected 'diagnosis' as the target column.

📊 Evaluation Metrics:
✅ Accuracy : 0.9649
✅ Precision: 0.9756
✅ Recall   : 0.9302
```



Shubham_Kumar_20240110030
colab.research.google.com/drive/1nFJG-2joxwenaaeuamjagAnthkFUhMtk#scrollTo=F0NZCdQ0lcr-
Shubham_Kumar_202401100300243.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text
Share
Gemini
RAM
Disk
Predicted
Patient Segmentation (PCA + KMeans)
Cluster
0
1
2
PCA 2
PCA 1
10
5
0
−5
−5
0
5
10
15
Activate Windows
Go to Settings to activate Windows.
16s completed at 2:32 PM
Type here to search
36°C
ENG
14:52
18-04-2025

```
✅ Precision: 0.9756
✅ Recall   : 0.9302
```



5. References & Credits

Libraries Used

- **Pandas** – Data manipulation
- **Scikit-learn** – Machine learning models
- **Seaborn/Matplotlib** – Visualizations

Tutorials & Resources

- Scikit-learn Documentation
- Kaggle AutoML Guides