## PROJECT 1: HR ANALYTICS

**Project Title: Data Cleaning and Visualization for HR Dataset**

**Objective**

Clean and visualize an HR dataset that includes intentional errors and inconsistencies. The goal is to prepare the data for trustworthy HR analytics and reporting.

Filename: Data Cleaning for HR Analytics v1.csv

Location: <https://github.com/swapnilsaurav/projects>

**Challenges Introduced**

* Placeholder values (e.g., "unknown", NaN)
* Invalid values (e.g., "not\_an\_email", negative salaries, unrealistic ages)
* Corrupt date fields
* Duplicate records
* Inconsistent categorical data

**Solution Code:**

|  |
| --- |
| import pandas as pd import numpy as np import matplotlib.pyplot as plt import seaborn as sns  file="https://raw.githubusercontent.com/swapnilsaurav/projects/refs/heads/main/Data%20Cleaning%20for%20HR%20Analytics%20v1.csv" *# Load updated dataset* df = pd.read\_csv(file)  *# Step 1: Remove duplicates* df = df.drop\_duplicates()  *# Step 2: Handle missing or invalid data* df["Name"] = df["Name"].fillna("Unknown")  df["Email"] = df["Email"].replace(["unknown", "not\_an\_email"], np.nan) df["Email"] = df["Email"].fillna(method="bfill")  df["Age"] = pd.to\_numeric(df["Age"], errors='coerce') df["Age"] = df["Age"].apply(lambda x: x if 18 <= x <= 70 else np.nan) df["Age"] = df["Age"].fillna(df["Age"].median())  df["Salary"] = pd.to\_numeric(df["Salary"], errors='coerce') df["Salary"] = df["Salary"].apply(lambda x: x if x >= 0 else np.nan) df["Salary"] = df["Salary"].fillna(df["Salary"].mean())  df["JoinDate"] = pd.to\_datetime(df["JoinDate"], errors='coerce')  df["Department"] = df["Department"].fillna("Not Specified") df["Remarks"] = df["Remarks"].replace("", np.nan).fillna("No Remarks")  *# Step 3: Treat Outliers* Q1 = df["Salary"].quantile(0.25) Q3 = df["Salary"].quantile(0.75) IQR = Q3 - Q1 lower\_bound, upper\_bound = Q1 - 1.5 \* IQR, Q3 + 1.5 \* IQR df["Salary"] = np.clip(df["Salary"], lower\_bound, upper\_bound)  *# Step 4: Visualizations  # 1. Age Distribution* plt.figure(figsize=(8, 4)) sns.histplot(df["Age"], bins=30, kde=True) plt.title("Age Distribution") plt.xlabel("Age") plt.ylabel("Frequency") plt.show()  *# 2. Salary Distribution* plt.figure(figsize=(8, 4)) sns.boxplot(x=df["Salary"]) plt.title("Salary Distribution with Outliers Treated") plt.xlabel("Salary") plt.show()  *# 3. Department Counts* plt.figure(figsize=(8, 4)) sns.countplot(y="Department", data=df, order=df["Department"].value\_counts().index) plt.title("Employees per Department") plt.xlabel("Count") plt.ylabel("Department") plt.show()  *# 4. Join Date Trend* plt.figure(figsize=(10, 4)) df["JoinYear"] = df["JoinDate"].dt.year sns.countplot(x="JoinYear", data=df.sort\_values("JoinYear")) plt.title("Employee Joining Trend by Year") plt.xlabel("Year") plt.ylabel("Count") plt.xticks(rotation=45) plt.show() |

**Visualizations Include:**

1. Age Distribution – To observe age spread and outlier treatment.
2. Salary Boxplot – Highlights the IQR method used to cap outliers.
3. Department Count – Shows HR distribution.
4. Join Date Trend – Annual hiring trends.

## PROJECT 2: HEALTHCARE DATA ANALYTICS

**Project Title: Preparing Patient Records for Clinical Analysis**

**Problem Statement**

A multi-specialty hospital has compiled patient records from different departments into a central system. However, the dataset has multiple issues:

* Missing values for key attributes like age, diagnosis, and gender
* Inconsistent formats for dates and medical condition names
* Placeholder or junk entries such as "unknown", "N/A", or "-"
* Unrealistic or outlier values in metrics such as blood pressure or age
* Duplicate records due to system sync errors

The data team must clean this dataset to enable accurate health analytics such as comorbidity detection, patient demographics profiling, and departmental load analysis.

**Dataset:** healthcare\_patient\_data.csv

**Location:** <https://github.com/swapnilsaurav/projects>

**Objective**

Use **Pandas**, **Matplotlib**, and **Seaborn** to:

1. Detect and correct missing or invalid entries
2. Remove or impute corrupt and inconsistent data
3. Normalize categorical fields like gender and diagnosis
4. Handle date formatting issues
5. Visualize key patient metrics for insights

**Sample Dataset Columns**

| **Column** | **Description** |
| --- | --- |
| PatientID | Unique patient identifier |
| Name | Patient's full name |
| Gender | Gender (can be inconsistent: 'M', 'Male', 'F', 'Female', etc.) |
| Age | Patient age (some unrealistic or missing) |
| VisitDate | Date of visit (some invalid entries) |
| Department | Department visited (e.g., Cardiology, ENT, etc.) |
| Diagnosis | Primary diagnosis (may contain variations/spelling errors) |
| BloodPressure | Recorded blood pressure (format issues / outliers) |
| Notes | Free-text doctor notes (may be blank or junk) |

**Solution Code**

|  |
| --- |
| import pandas as pd import numpy as np import matplotlib.pyplot as plt import seaborn as sns  file="https://raw.githubusercontent.com/swapnilsaurav/projects/refs/heads/main/healthcare\_patient\_data.csv" *# Load Dataset* df = pd.read\_csv(file)  *# Step 1: Drop duplicate records* df = df.drop\_duplicates()  *# Step 2: Normalize Gender values* df["Gender"] = df["Gender"].str.strip().str.upper().replace({  "MALE": "M", "FEMALE": "F", "UNKNOWN": np.nan, "-": np.nan }) df["Gender"] = df["Gender"].fillna("Other")  *# Step 3: Fix Age values* df["Age"] = pd.to\_numeric(df["Age"], errors='coerce') df["Age"] = df["Age"].apply(lambda x: x if 0 < x < 120 else np.nan) df["Age"] = df["Age"].fillna(df["Age"].median())  *# Step 4: Fix VisitDate* df["VisitDate"] = pd.to\_datetime(df["VisitDate"], errors='coerce')  *# Step 5: Clean Blood Pressure # Convert string like '120/80' to systolic and diastolic* bp\_split = df["BloodPressure"].str.extract(r'(?P<Systolic>\d{2,3})/(?P<Diastolic>\d{2,3})') df["Systolic"] = pd.to\_numeric(bp\_split["Systolic"], errors='coerce') df["Diastolic"] = pd.to\_numeric(bp\_split["Diastolic"], errors='coerce') df["Systolic"] = df["Systolic"].clip(90, 200) df["Diastolic"] = df["Diastolic"].clip(60, 120)  *# Step 6: Normalize Diagnoses* df["Diagnosis"] = df["Diagnosis"].str.strip().str.title() df["Diagnosis"] = df["Diagnosis"].replace({"N/A": np.nan, "-": np.nan}) df["Diagnosis"] = df["Diagnosis"].fillna("Not Specified")  *# Step 7: Clean Notes* df["Notes"] = df["Notes"].replace(["", "-", "N/A", "unknown"], np.nan) df["Notes"] = df["Notes"].fillna("No additional notes.")  *# Step 8: Visualizations  # Age Distribution* plt.figure(figsize=(8, 4)) sns.histplot(df["Age"], bins=30, kde=True) plt.title("Age Distribution of Patients") plt.xlabel("Age") plt.ylabel("Count") plt.show()  *# Gender Distribution* plt.figure(figsize=(6, 4)) sns.countplot(x="Gender", data=df) plt.title("Gender Breakdown") plt.xlabel("Gender") plt.ylabel("Patient Count") plt.show()  *# Department Count* plt.figure(figsize=(10, 4)) sns.countplot(y="Department", data=df, order=df["Department"].value\_counts().index) plt.title("Patient Visits per Department") plt.xlabel("Count") plt.ylabel("Department") plt.show()  *# Blood Pressure - Boxplots* plt.figure(figsize=(8, 4)) sns.boxplot(data=df[["Systolic", "Diastolic"]]) plt.title("Blood Pressure Distribution") plt.ylabel("Pressure (mm Hg)") plt.show()  *# Visit Trend Over Time* plt.figure(figsize=(10, 4)) df["VisitMonth"] = df["VisitDate"].dt.to\_period("M") sns.countplot(x="VisitMonth", data=df.sort\_values("VisitMonth")) plt.xticks(rotation=45) plt.title("Patient Visits Over Time") plt.xlabel("Visit Month") plt.ylabel("Visit Count") plt.show() |

**Final Outcome**

After cleaning:

* All genders and diagnoses are standardized
* Invalid ages and dates are corrected
* Blood pressure data is split and validated
* Missing text data is handled gracefully
* Key metrics are visualized for insights

\* \* \*