Task 1 Mokshank Vinod: Data Loading & Cleaning and Preprocessing import pandas as pd In [1]: import numpy as np df = pd.read_csv(r"C:\Users\rizwa\OneDrive\Desktop\Intern\Final project\HR-Employee-Attrition.csv") print("Data Information:") print(df.info()) # Check for missing values print("\nMissing Values:") print(df.isnull().sum()) # Handle missing values (if any) df = df.dropna() # Display basic statistics after handling missing values print("\nSummary Statistics:") print(df.describe()) # Check for duplicates print("\nDuplicate Rows:") print(df.duplicated().sum()) # Handle duplicates (if any) # Example: Drop duplicate rows df = df.drop_duplicates() # Confirm changes print("\nData Information After Cleaning:") print(df.info()) Data Information: <class 'pandas.core.frame.DataFrame'> RangeIndex: 1470 entries, 0 to 1469 Data columns (total 35 columns): Column # Non-Null Count Dtype -------------1470 non-null int64 0 Age 1 Attrition 1470 non-null object BusinessTravel 1470 non-null object DailyRate 1470 non-null 3 int64 Department 1470 non-null object 1470 non-null DistanceFromHome int64 Education 1470 non-null EducationField 1470 non-null object EmployeeCount 1470 non-null 8 int64 1470 non-null EmployeeNumber 9 int64 10 EnvironmentSatisfaction 1470 non-null int64 11 Gender 1470 non-null object 1470 non-null 12 HourlyRate int64 1470 non-null 13 JobInvolvement int64 JobLevel 1470 non-null int64 14 JobRole 1470 non-null 15 object 16 JobSatisfaction 1470 non-null int64 17 MaritalStatus 1470 non-null object MonthlyIncome 1470 non-null 18 int64 MonthlyRate 1470 non-null 19 int64 20 NumCompaniesWorked 1470 non-null int64 21 0ver18 1470 non-null object 22 OverTime 1470 non-null object 1470 non-null 23 PercentSalaryHike 24 PerformanceRating 1470 non-null int64 RelationshipSatisfaction 1470 non-null 25 int64 1470 non-null 26 StandardHours int64 27 StockOptionLevel 1470 non-null int64 28 TotalWorkingYears 1470 non-null int64 29 TrainingTimesLastYear 1470 non-null int64 30 WorkLifeBalance 1470 non-null int64 31 YearsAtCompany 1470 non-null int64 YearsInCurrentRole 1470 non-null 33 YearsSinceLastPromotion 1470 non-null 34 YearsWithCurrManager 1470 non-null dtypes: int64(26), object(9) memory usage: 402.1+ KB Missing Values: 0 Age Attrition 0 BusinessTravel 0 DailyRate 0 Department 0 ${\tt DistanceFromHome}$ 0 Education 0 EducationField 0 EmployeeCount 0 EmployeeNumber 0 EnvironmentSatisfaction 0 Gender 0 HourlyRate 0 JobInvolvement JobLevel 0 JobRole 0 JobSatisfaction 0 MaritalStatus 0 MonthlyIncome 0 MonthlyRate 0 NumCompaniesWorked 0 0ver18 0 OverTime 0 PercentSalaryHike 0 PerformanceRating 0 ${\tt RelationshipSatisfaction}$ 0 StandardHours 0 StockOptionLevel TotalWorkingYears 0 TrainingTimesLastYear 0 WorkLifeBalance YearsAtCompany 0 YearsInCurrentRole 0 YearsSinceLastPromotion YearsWithCurrManager dtype: int64 **Summary Statistics:** Age DailyRate DistanceFromHome Education EmployeeCount \ 1470.000000 1470.000000 1470.000000 1470.000000 1470.0 count 36.923810 802.485714 9.192517 2.912925 1.0 mean 9.135373 8.106864 std 403.509100 1.024165 0.0 18.000000 102.000000 1.000000 1.000000 1.0 min 25% 30.000000 465.000000 2.000000 2.000000 1.0 50% 36.000000 802.000000 7.000000 3.000000 1.0 43.000000 14.000000 75% 1157.000000 4.000000 1.0 max 60.000000 1499.000000 29.000000 5.000000 1.0 HourlyRate JobInvolvement \ EmployeeNumber EnvironmentSatisfaction 1470.000000 1470.000000 1470.000000 1470.000000 count mean 1024.865306 2.721769 65.891156 2.729932 602.024335 1.093082 20.329428 0.711561 std 30.000000 1.000000 1.000000 1.000000 min 25% 491.250000 2.000000 48.000000 2.000000 50% 1020.500000 3.000000 66.000000 3.000000 75% 1555.750000 4.000000 83.750000 3.000000 2068.000000 4.000000 100.000000 4.000000 max JobLevel ... RelationshipSatisfaction StandardHours \ 1470.0 count 1470.000000 ... 1470.000000 80.0 mean 2.063946 ... 2.712245 1.106940 ... 1.081209 std 0.0 min 1.000000 1.000000 0.08 1.000000 25% 2.000000 80.0 2.000000 ... 50% 3.000000 80.0 75% 4.000000 3.000000 80.0 max 5.000000 4.000000 80.0 TotalWorkingYears TrainingTimesLastYear \ StockOptionLevel 1470.000000 1470.000000 1470.000000 count 0.793878 11.279592 2.799320 mean 0.852077 7.780782 1.289271 std 0.000000 0.00000 0.000000 min 2.000000 25% 0.000000 6.000000 50% 10.000000 3.000000 1.000000 75% 15.000000 3.000000 1.000000 3.000000 6.000000 max 40.000000 WorkLifeBalance YearsAtCompany YearsInCurrentRole \ count 1470.000000 1470.000000 1470.000000 2.761224 7.008163 4.229252 mean 0.706476 6.126525 3.623137 std min 1.000000 0.00000 0.000000 25% 3.000000 2.000000 2.000000 50% 3.000000 5.000000 3.000000 75% 3.000000 9.000000 7.000000 max 4.000000 40.000000 18.000000 YearsSinceLastPromotion YearsWithCurrManager 1470.000000 count 1470.000000 2.187755 4.123129 mean 3.222430 3.568136 std min 0.000000 0.000000 25% 0.000000 2.000000 50% 1.000000 3.000000 3.000000 75% 7.000000 15.000000 17.000000 max [8 rows x 26 columns] Duplicate Rows: Data Information After Cleaning: <class 'pandas.core.frame.DataFrame'> Int64Index: 1470 entries, 0 to 1469 Data columns (total 35 columns): # Column Non-Null Count Dtype -----0 Age 1470 non-null int64 Attrition 1470 non-null object 1 2 BusinessTravel 1470 non-null object 1470 non-null 3 DailyRate int64 4 Department 1470 non-null object 5 DistanceFromHome 1470 non-null int64 6 Education 1470 non-null int64 7 1470 non-null object EducationField 1470 non-null 8 EmployeeCount int64 9 **EmployeeNumber** 1470 non-null int64 10 EnvironmentSatisfaction 1470 non-null int64 11 Gender 1470 non-null object HourlyRate 12 1470 non-null int64 13 1470 non-null int64 JobInvolvement 14 JobLevel 1470 non-null int64 15 JobRole 1470 non-null object JobSatisfaction 1470 non-null 17 MaritalStatus 1470 non-null object MonthlyIncome 1470 non-null MonthlyRate 19 1470 non-null int64 20 NumCompaniesWorked 1470 non-null int64 21 0ver18 1470 non-null object 22 OverTime 1470 non-null object 23 PercentSalaryHike 1470 non-null int64 24 PerformanceRating 1470 non-null int64 25 RelationshipSatisfaction 1470 non-null int64 StandardHours int64 26 1470 non-null 27 1470 non-null int64 StockOptionLevel 28 TotalWorkingYears 1470 non-null int64 29 TrainingTimesLastYear 1470 non-null int64 1470 non-null 30 WorkLifeBalance int64 31 YearsAtCompany 1470 non-null int64 YearsInCurrentRole 1470 non-null 33 YearsSinceLastPromotion 1470 non-null int64 34 YearsWithCurrManager 1470 non-null int64 dtypes: int64(26), object(9) memory usage: 413.4+ KB Task 2 RIzwan Siddiqui: Data Analysis and Modeling (Analyzed the cleaned HR data to identify patterns, trends, and correlations, EDA: Exploratory Data Analysis) import pandas as pd In [11]: import matplotlib.pyplot as plt import seaborn as sns # Visualize the distribution of Age plt.figure(figsize=(10, 6)) sns.histplot(df['Age'], bins=30, kde=True) plt.title('Distribution of Age') plt.show() Distribution of Age 140 120 100 80 60 40 20 30 40 20 50 Age In [12]: # Visualize the attrition count plt.figure(figsize=(8, 5)) sns.countplot(x='Attrition', data=df) plt.title('Attrition Count') plt.show() Attrition Count 1200 1000 800 count 600 400 200 Yes No Attrition In [13]: # Explore the correlation matrix correlation_matrix = df.corr() plt.figure(figsize=(12, 10)) sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5) plt.title('Correlation Matrix') plt.show() C:\Users\rizwa\AppData\Local\Temp\ipykernel_1516\4129031465.py:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will def ault to False. Select only valid columns or specify the value of numeric_only to silence this warning. correlation_matrix = df.corr() Correlation Matrix 1.0 .010.01).0240.03<mark>0.51</mark>.004<mark>0.5</mark>0.028**0.3**.00060011905 0.010.0010.2103:<mark>0.68</mark>0.040.02 0.310.210.22 0.2 DailyRate -.011 1-0.095.01 .05q1.01q8.02q8.04q6.00q8.0301.007q7.03q203q8.0q2.\$00q04q707 04Ql.010500z0503QBl.018H009Q903QBl.02 DistanceFromHome -.00-07005 1 0.02 0405004160307.02017009150190.010.01 036.01050301006480061300-\$070107.02-07.02 10.040.02017006 Education -0.210.010.021 01**(0.15**().02/5009(806(0.06().05(4.06 EmployeeCount -- 0.8 EmployeeNumber - 0.0-0.051036.04 1 0.0108.0305.00-00903.0405.0305.0308.00-0301-10.020.0 0.6pl.01040240.0±0.0−101.00+840+019009 .01**8 1 -**0.905,000\$1900µ2090\$180**0**|303**(**\$.01-(\$.03-|2).005\007 EnvironmentSatisfaction -0.010.018.016.02)0-6|400-20701020208001050108.01-61.00 .0-15.0-0-21.30-0-28.500-405.0-20.02-(41.02-10.02 HourlyRate -. 024.026.030.01 0 **601, 9**0 **8**) 30 4 **3 1 -** 0, 0 £012, 0 2 **0**, 0 £05, 0 £05, 0 £0 £0, 0 £07, 0 2 **0**, 0 3 0=202:00-505:01:05:01:00:201:00=807:0201+02 JobInvolvement -0.030.04050080804 - 0.6 JobLevel -0.510.000800530.1 780.01@03(0.530.390.350.38 .0-40600-4080741.0-201001<mark>91-0</mark>.0**0**.70200-40510.02100-20801 01+10.002.00-5080-1090001.800-2030148.02 JobSatisfaction -.0009030.00-0701 MonthlyIncome - 0.5 00700107.09 0.020203 <mark>0.51</mark>0.360.340.34 MonthlyRate -.028.032020.02 .01\$.03\$.01\$.01\$.004000\$.403510.0408096.4099.800 .0344.0206001050048.0244.01030030603 NumCompaniesWorked - 0.30.038.028 0.140.05<mark>0.15</mark>0.01**8 1 -**0.0-10.0144.05 .03<mark>0.24</mark>9.0606008G4.149.0901.0340.1 001030103.0202.015 - 0.4 PercentSalaryHike --00360230.040.01 .01-(31.0-201200-40)101-(07.03|5.02(0.0-20)7006|41.01)07050-2010905/200-3030-30600-1050202.01 0.02/0.904.00-202020p3.02010020p30-101700-40801 PerformanceRating --000**1/9**00/4/10/2/10/02)0G|5006b7.0D|600Q|6003B403(\$.01&k.02 0.405.020400215.020.01-0.0105.0013000 .001/00/01/00/01/30344.0242.01/02/02/05/00/40/1052/0.04/.03/1.1 StandardHours -StockOptionLevel -.03&040.046.01 060200344.050.0202.0144.0101.00504.0349.001.0007150031504 .0**1)**.0**1**01004**)**101**6**.05**(**1.014.02) - 0.2 .0-1040.602.70.602.800.505.780.020.770.02(0.24).02010060702 1-0.03600 <mark>0.63</mark>0.46 0.4 0.46 TotalWorkingYears -0.68 .0150040.15 TrainingTimesLastYear -0.002002050307.02 .02-ф.0-10900-80501-бs.0-10800-508020200-1050-6600-5020108002 01-01.03 6 1 0.020 80 0 18 60 9 3 10 9 0 10 0 00401.00**0**.02**8 1 0**.0120.0**5**.00**8**9002 .01¢.050800-ф601¢503-ф.01¢803¢1.0508090\$,⊕004,502¢5.02 WorkLifeBalance -0.020.038.007009 YearsAtCompany -0.310.084009506 .0101001-16.0-20.02<mark>0.53</mark>.003<mark>0.51</mark>).02-19.1-20.013130030401 .01:0.60.0036012 1 760.620. - 0.0 0084018.024008<mark>0.34</mark>.002<mark>0.36</mark>).0148.049100105036.01 05:0.46.0050.050.7 YearsInCurrentRole -0.20 00990190.0 6 1 0.550.71 .00\$20**1\$**.02\$7.02\$<mark>0.35</mark>).01<mark>0.34</mark>003\$1603\$7.02\$201\$.03 .0140.40 .002 008 0.62 0.55 YearsSinceLastPromotion -0.220.038.010.05 YearsWithCurrManager - 0.20.025014.0 JobLevel YearsInCurrentRole DistanceFromHome Education EmployeeNumber Jobinvolvement JobSatisfaction **NumCompaniesWorked** StockOptionLevel TotalWorkingYears TrainingTimesLastYear WorkLifeBalance YearsSinceLastPromotion YearsWithCurrManager DailyRate EmployeeCount EnvironmentSatisfaction HourlyRate MonthlyRate **PercentSalaryHike RelationshipSatisfaction** StandardHours YearsAtCompany MonthlyIncome PerformanceRating In [15]: # Explore relationships between specific variables plt.figure(figsize=(12, 6)) sns.boxplot(x='JobRole', y='MonthlyIncome', data=df) plt.title('Monthly Income by Job Role') plt.xticks(rotation=45, ha='right') plt.show() Monthly Income by Job Role 20000 17500 15000 MonthlyIncome 12500 10000 7500 5000 2500 JobRole ### Trends in Salary Over Time ### plt.figure(figsize=(12, 6)) sns.lineplot(data=df, x='YearsAtCompany', y='MonthlyIncome', hue='Attrition') plt.title('Trends in Salary Over Time') plt.xlabel('Years at Company') plt.ylabel('Monthly Income') plt.show() Trends in Salary Over Time 20000 Attrition Yes No 17500 15000 Monthly Income 12500 10000 7500 5000 2500 10 15 20 25 30 35 40 5 Years at Company In [8]: ### Patterns in Performance Ratings ### plt.figure(figsize=(10, 6)) sns.countplot(data=df, x='PerformanceRating', hue='Attrition') plt.title('Patterns in Performance Ratings') plt.xlabel('Performance Rating') plt.ylabel('Count') plt.show() Patterns in Performance Ratings Attrition 1000 Yes No 800 600 400 200 Performance Rating In [9]: ### Trends in Turnover Over Time ### plt.figure(figsize=(12, 6)) sns.lineplot(data=df, x='YearsAtCompany', y='Attrition', estimator='mean', ci=None) plt.title('Trends in Turnover Over Time') plt.xlabel('Years at Company') plt.ylabel('Attrition Rate') plt.show() C:\Users\rizwa\AppData\Local\Temp\ipykernel_1516\3951006875.py:3: FutureWarning: The `ci` parameter is deprecated. Use `errorbar=None` for the same effect. sns.lineplot(data=df, x='YearsAtCompany', y='Attrition', estimator='mean', ci=None) Trends in Turnover Over Time Yes Attrition Rate No 5 10 15 20 25 30 35 40 Years at Company In [10]: ### Correlation Between Job Satisfaction and Turnover ### plt.figure(figsize=(10, 6)) sns.boxplot(data=df, x='Attrition', y='JobSatisfaction') plt.title('Correlation Between Job Satisfaction and Turnover') plt.xlabel('Attrition') plt.ylabel('Job Satisfaction') plt.show() Correlation Between Job Satisfaction and Turnover 4.0 3.5 3.0 Satisfac 2.5 2.0 1.5 1.0 Yes No Attrition Task 3 Rizwan Siddiqui: Strategy Development and Documentation In [23]: **class** Employee: def __init__(self, name, position, skills, performance_rating): self.name = nameself.position = position self.skills = skills self.performance_rating = performance_rating self.training_completed = False def set_performance_rating(self, rating): self.performance_rating = rating def complete_training(self): self.training_completed = True print(f"{self.name} has completed targeted training.") def adjust_compensation(self, new_salary): print(f"Compensation for {self.name} adjusted to {new_salary}.") def promote(self, new_position): print(f"{self.name} has been promoted to {new_position}.") self.position = new_position class PerformanceEnhancementStrategy: def __init__(self): self.employees = [] def add_employee(self, employee): self.employees.append(employee) def implement_strategy(self): for employee in self.employees: if employee.performance_rating < 4.0:</pre> # Offer targeted training for employees with low performance employee.complete_training() elif 4.0 <= employee.performance_rating < 4.5:</pre> # Consider compensation adjustment for employees with moderate performance employee.adjust_compensation(1.05) elif employee.performance_rating >= 4.5: # Consider career progression for high-performing employees employee.promote("Senior " + employee.position) employee1 = Employee("Ekta Agrawal", "Software Developer", ["Python", "JavaScript"], 3.8) employee2 = Employee("Harshvardhan", "Data Analyst", ["SQL", "Excel"], 4.2) employee3 = Employee("Rizwan Siddiqui", "UX Designer", ["UI/UX", "Sketch"], 4.8) strategy = PerformanceEnhancementStrategy() strategy.add_employee(employee1) strategy.add_employee(employee2) strategy.add_employee(employee3) # Implement the performance enhancement strategy strategy.implement_strategy() # Print the results for employee in strategy.employees: print(f"{employee.name} - Position: {employee.position}, Performance Rating: {employee.performance_rating}") Ekta Agrawal has completed targeted training. Compensation for Harshvardhan adjusted to 1.05. Rizwan Siddigui has been promoted to Senior UX Designer. Ekta Agrawal - Position: Software Developer, Performance Rating: 3.8 Harshvardhan - Position: Data Analyst, Performance Rating: 4.2 Rizwan Siddiqui - Position: Senior UX Designer, Performance Rating: 4.8