Employee Salaries Analysis

Using python , pandas , numpy and Jupyter Notebook

Discription :

This project focuses on performing exploratory data analysis (EDA) on a salary dataset using Python. The objectives included importing CSV data, cleaning and formatting it, and deriving useful statistics and insights.

- ✅ Imported the dataset using pandas.read_csv()
- ✅ Displayed top/bottom rows, column types, and data structure
- ✅ Applied .info(), .describe() for quick data summary
- ✅ Practiced real-world EDA workflows

Learning Outcome: Strengthened my understanding of data analysis, Python libraries, and the Jupyter workflow essential for data science.

jupyter Employee Salaries Last Checkpoint: 3 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[166]:
```
import pandas as pd
import numpy as np
```

importing csv files

[167]:
```
data = pd.read_csv("Salaries.csv",low_memory=False)
```

1. Display Top 5 Rows of The Dataset

[144]:
```
data.head(5)
```

[144]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.0	400184.25	NaN	567595.43	567595.43	2011	NaN	San Francisco	NaN
1	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	NaN	538909.28	538909.28	2011	NaN	San Francisco	NaN
2	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.6	NaN	335279.91	335279.91	2011	NaN	San Francisco	NaN
3	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.0	56120.71	198306.9	NaN	332343.61	332343.61	2011	NaN	San Francisco	NaN
4	5	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT, (FIRE DEPARTMENT)	134401.6	9737.0	182234.59	NaN	326373.19	326373.19	2011	NaN	San Francisco	NaN

2.Check Last 5 Rows of The Dataset

[43]:
```
data.tail(5)
```

[43]:

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status
148649	148650	Roy I Tillery	Custodian	0.00	0.00	0.00	0.00	0.00	0.00	2014	NaN	San Francisco	PT
148650	148651	Not provided	Not provided	Not Provided	Not Provided	Not Provided	Not Provided	0.00	0.00	2014	NaN	San Francisco	NaN
148651	148652	Not provided	Not provided	Not Provided	Not Provided	Not Provided	Not Provided	0.00	0.00	2014	NaN	San Francisco	NaN
148652	148653	Not provided	Not provided	Not Provided	Not Provided	Not Provided	Not Provided	0.00	0.00	2014	NaN	San Francisco	NaN
148653	148654	Joe Lopez	Counselor, Log Cabin Ranch	0.00	0.00	-618.13	0.00	-618.13	-618.13	2014	NaN	San Francisco	PT

3. Find Shape of Our Dataset (Number of Rows And Number of Columns)

[44]:
```
data.shape
```

[44]:
```
(148654, 13)
```

4. Getting Information About Our Dataset Like Total Number Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement

[45]:
```
#total number of rows
len(data)
```

[45]:
```
148654
```

[46]:
```
#total number of columns
len(data.columns)
```

[46]:
```
13
```

[47]:
```
#datatypes
data.dtypes
```

[47]:
```
Id                    int64
EmployeeName         object
JobTitle             object
BasePay              object
OvertimePay          object
OtherPay             object
Benefits             object
TotalPay            float64
TotalPayBenefits    float64
Year                  int64
Notes               float64
Agency               object
Status               object
dtype: object
```

[48]:

```python
#Memory Requirement
data.info(memory_usage="deep")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148654 entries, 0 to 148653
Data columns (total 13 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Id                148654 non-null  int64
 1   EmployeeName      148654 non-null  object
 2   JobTitle          148654 non-null  object
 3   BasePay           148049 non-null  object
 4   OvertimePay       148654 non-null  object
 5   OtherPay          148654 non-null  object
 6   Benefits          112495 non-null  object
 7   TotalPay          148654 non-null  float64
 8   TotalPayBenefits  148654 non-null  float64
 9   Year              148654 non-null  int64
 10  Notes             0 non-null       float64
 11  Agency            148654 non-null  object
 12  Status            38119 non-null   object
dtypes: float64(3), int64(2), object(8)
memory usage: 68.9 MB
```

5. Check Null Values In The Dataset

[58]:
```
data.isnull().sum()
```

[58]:
```
Id                     0
EmployeeName           0
JobTitle               0
BasePay              605
OvertimePay            0
OtherPay               0
Benefits           36159
TotalPay               0
TotalPayBenefits       0
Year                   0
Notes             148654
Agency                 0
Status            110535
dtype: int64
```

6.Drop ID, Notes, Agency, and Status Columns

[75]:
```
df = data.drop(["Id","Notes","Agency","Status"],axis = 1)
df
```

[75]:

	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year
0	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.0	400184.25	NaN	567595.43	567595.43	2011
1	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	NaN	538909.28	538909.28	2011
2	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.6	NaN	335279.91	335279.91	2011
3	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.0	56120.71	198306.9	NaN	332343.61	332343.61	2011
4	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)	134401.6	9737.0	182234.59	NaN	326373.19	326373.19	2011
...	...	...	...	...	...	...	...	...	...
148649	Roy I Tillery	Custodian	0.00	0.00	0.00	0.00	0.00	0.00	2014
148650	Not provided	Not provided	Not Provided	Not Provided	Not Provided	Not Provided	0.00	0.00	2014
148651	Not provided	Not provided	Not Provided	Not Provided	Not Provided	Not Provided	0.00	0.00	2014
148652	Not provided	Not provided	Not Provided	Not Provided	Not Provided	Not Provided	0.00	0.00	2014
148653	Joe Lopez	Counselor, Log Cabin Ranch	0.00	0.00	-618.13	0.00	-618.13	-618.13	2014

7. Get Overall Statistics About The Dataframe

[59]:
```
data.describe()
```

[59]:

	Id	TotalPay	TotalPayBenefits	Year	Notes
count	148654.000000	148654.000000	148654.000000	148654.000000	0.0
mean	74327.500000	74768.321972	93692.554811	2012.522643	NaN
std	42912.857795	50517.005274	62793.533483	1.117538	NaN
min	1.000000	-618.130000	-618.130000	2011.000000	NaN
25%	37164.250000	36168.995000	44065.650000	2012.000000	NaN
50%	74327.500000	71426.610000	92404.090000	2013.000000	NaN
75%	111490.750000	105839.135000	132876.450000	2014.000000	NaN
max	148654.000000	567595.430000	567595.430000	2014.000000	NaN

8. Find Occurrence of The Employee Names (Top 5)

[66]:
```
employe = data["EmployeeName"].value_counts().head(5)
print(employe)
```

```
EmployeeName
Kevin Lee       13
William Wong    11
Richard Lee     11
Steven Lee      11
John Chan        9
Name: count, dtype: int64
```

9. Find The Number of Unique Job Titles

[71]:
```
uniNUm = data["JobTitle"].nunique()
print(uniNUm)
```

```
2159
```

10.Total Number of Job Titles Contain Captain

[87]:
```
len(data[data["JobTitle"].str.contains("Captain",case = False)])
```

[87]:
```
552
```

11. Display All the Employee Names From Fire Department

[91]:
```
data[data["JobTitle"].str.contains("Fire Department" , case = False)]["EmployeeName"]
```

[91]:
```
4              PATRICK GARDNER
6                    ALSON LEE
8               MICHAEL MORRIS
9           JOANNE HAYES-WHITE
10               ARTHUR KENNEY
                   ...
32623             JAMES BARDEN
36162       Joanne Hayes-White
72926     Joanne M Hayes-White
102303          Robert E Evans
110535    Joanne M Hayes-White
Name: EmployeeName, Length: 226, dtype: object
```

12. Find Minimum, Maximum, and Average BasePay

[95]:
```
#conerting in number
data['BasePay'] = pd.to_numeric(data['BasePay'], errors='coerce')
data["BasePay"].describe()
```

[95]:
```
count    148045.000000
mean      66325.448840
std       42764.635495
min        -166.010000
25%       33588.200000
50%       65007.450000
75%       94691.050000
max      319275.010000
Name: BasePay, dtype: float64
```

13. Replace 'Not Provided' in EmployeeName' Column to NaN `

[103]:
```
data["EmployeeName"] = data["EmployeeName"].replace("Not provided",np.nan)
data["EmployeeName"]
```

[103]:
```
0            NATHANIEL FORD
1              GARY JIMENEZ
2            ALBERT PARDINI
3         CHRISTOPHER CHONG
4           PATRICK GARDNER
                ...
148649        Roy I Tillery
148650                  NaN
148651                  NaN
148652                  NaN
148653            Joe Lopez
Name: EmployeeName, Length: 148654, dtype: object
```

14. Drop The Rows Having 5 Missing Values

[147]:
```
# Option 1: inplace=True ke saath (no need to assign back)
data.drop(data[data.isnull().sum(axis=1) == 5].index, axis=0, inplace=True)

# Option 2: inplace=False (default) aur assign karna
#data = data.drop(data[data.isnull().sum(axis=1) == 5].index, axis=0)
```

[148]:
```
data.isnull().sum()
```

[148]:
```
Id                     0
EmployeeName           0
JobTitle               0
BasePay              605
OvertimePay            0
OtherPay               0
Benefits           36159
TotalPay               0
TotalPayBenefits       0
Year                   0
Notes             148654
Agency                 0
Status            110535
dtype: int64
```

15. Find Job Title of ALBERT PARDINI

[154]:
```
mask = data["EmployeeName"].str.contains("ALBERT PARDINI" ,case = False,na = False)
job_titles = data.loc[mask ,"JobTitle"]
print(job_titles)
```

```
2        CAPTAIN III (POLICE DEPARTMENT)
36519                          Captain 3
Name: JobTitle, dtype: object
```

[156]:
```
data[data["EmployeeName"] == "ALBERT PARDINI"]["JobTitle"]
```

[156]:
```
2    CAPTAIN III (POLICE DEPARTMENT)
Name: JobTitle, dtype: object
```

16. How Much ALBERT PARDINI Make (Include Benefits)?

[158]:
```
data[data["EmployeeName"]== "ALBERT PARDINI"]["TotalPayBenefits"]
```

[158]:
```
2    335279.91
Name: TotalPayBenefits, dtype: float64
```

17.Display Name of The Person Having The Highest BasePay

[171]:
```
data["BasePay"] = pd.to_numeric(data["BasePay"],errors="coerce")
data[data["BasePay"].max() == data["BasePay"]]["EmployeeName"]
```

[171]:
```
72925    Gregory P Suhr
Name: EmployeeName, dtype: object
```

18.Find Average BasePay of All Employee Per Year

[176]:
```
data["BasePay"] = pd.to_numeric(data["BasePay"], errors='coerce')
data.groupby("Year")["BasePay"].mean()
```

[176]:
```
Year
2011    63595.956517
2012    65436.406857
2013    69630.030216
2014    66564.421924
Name: BasePay, dtype: float64
```

19. Find Average BasePay of All Employee Per JobTitle

[177]:
```
data.groupby("JobTitle")["BasePay"].mean()
```

[177]:
```
JobTitle
ACCOUNT CLERK                                      43300.806506
ACCOUNTANT                                         46643.172000
ACCOUNTANT INTERN                                  28732.663958
ACPO,JuvP, Juv Prob (SFERS)                        62290.780000
ACUPUNCTURIST                                      66374.400000
                                                       ...
X-RAY LABORATORY AIDE                              47664.773077
X-Ray Laboratory Aide                              46086.387100
YOUTH COMMISSION ADVISOR, BOARD OF SUPERVISORS     52609.910000
Youth Comm Advisor                                 39077.957500
ZOO CURATOR                                        43148.000000
Name: BasePay, Length: 2159, dtype: float64
```

20. Find Average BasePay of Employee Having Job Title ACCOUNTANT

[181]:
```
data[data["JobTitle"] == "ACCOUNTANT"]["BasePay"].mean()
```

[181]:
```
np.float64(46643.172)
```

Find Top 5 Most Common Jobs

[182]:
```
data["JobTitle"].value_counts().head(5)
```

[182]:
```
JobTitle
Transit Operator                7036
Special Nurse                   4389
Registered Nurse                3736
Public Svc Aide-Public Works    2518
Police Officer 3                2421
Name: count, dtype: int64
```

ThankYou.....