1.01. data_cleaning

October 14, 2024

1 Project Intro / Objective

A newly established business school is exploring whether a data-driven approach to their admissions process can be more effective that their current method of using intuition. By adopting a data-driven strategy, the school hope to uncover patterns in student admissions that can improve decision-making and lead to better post-graduation outcomes, such as higher job placement rates.

The school has provided data from its first graduating class, including information on undergraduate degrees, MBA performance, work experience, and employment status two months after graduation. The objective of this project is to identify key factors that strongly correlate with post-graduation employment. These insights could be used to refine the admissions process, allowing the school to admit candidates with a higher likelihood of success, resulting in better graduation rates and job placement outcomes.

A data-drive approach has the potential to improve the efficiency of the admissions process but also positions the school as a forward-thinking and prestigious institution. The ultimate goal is to ensure continuing success for future graduates and to enhance the school's reputation as a trusted destination for aspiring MBA candidates.

1.1 Project Description

This project aims to determine if there are identifiable factors that can predict job placement success for MBA graduates. By analyzing data provided on the business school's first graduating class, the project will determine key indicators that correlate with job placement within two months of graduation.

2 Data Cleaning and Preprocessing

[6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import httpimport

from pathlib import Path
```

[7]:
```
# Import personal library
with httpimport.github_repo("junclemente", "jcds", ref="master"):
    import jcds.eda as jq
```

[8]:
```
datasets = Path("../datasets/")
df_file = "maven_business_school/MavenBusinessSchool(final).xlsx"
df = pd.read_excel(datasets / df_file)
df.head()
```

[8]:

	Student ID	Undergrad Degree	Undergrad Grade	MBA Grade	Work Experience	Employability (Before)	Employability (After)	Status	Annual Salary
0	1	Business	68.4	90.2	No	252.0	276.0	Placed	111000.0
1	2	Business	62.1	92.8	No	423.0	410.0	Not Placed	NaN
2	3	Computer Science	70.2	68.7	Yes	101.0	119.0	Placed	107000.0
3	4	Engineering	75.1	80.7	No	288.0	334.0	Not Placed	NaN
4	5	Finance	60.9	74.9	No	248.0	252.0	Not Placed	NaN

2.1 Data Quality Reports

[9]:
```
jq.quick_report(df)
```

```
==========================================
Quick Report - info(memory_usage='deep')
Total cols: 9
Rows missing all values: 0 (0.0%)
Total Rows: 1200
Cols with missing values: 4 (44.44%)
Total missing values in dataset: 4
==========================================
Categorical features: 3
- Undergrad Degree: 5 unique values
- Work Experience: 2 unique values
- Status: 2 unique values
==========================================
Continuous features: 6
- Student ID: 1200 unique values
- Undergrad Grade: 762 unique values
- MBA Grade: 837 unique values
- Employability (Before): 1182 unique values
- Employability (After): 776 unique values
- Annual Salary: 89 unique values
```

2.1.1 Categorical Features

[10]:
```
jq.dqr_cat(df)
```

```
The categorical features are:
['Undergrad Degree', 'Work Experience', 'Status']
Data Quality Report for Categorical Features
Total features: 3 / 1200 rows
==========================================
Stats
-----

            Feature  Count  Missing  % Missing  Cardinality
0  Undergrad Degree   1200        0        0.0            5
1   Work Experience   1200        0        0.0            2
2            Status   1200        0        0.0            2



Mode 1
------

            Feature    Mode 1  Mode 1 Freq.  Mode 1 %
0  Undergrad Degree  Business           257     21.42
1   Work Experience        No          1066     88.83
2            Status    Placed           716     59.67



Mode 2
------

            Feature            Mode 2  Mode 2 Freq.  Mode 2 %
0  Undergrad Degree  Computer Science           240     20.00
1   Work Experience               Yes           134     11.17
2            Status        Not Placed           484     40.33



Descriptive Stats
-----------------

                 count unique       top  freq
Undergrad Degree  1200      5  Business   257
Work Experience   1200      2        No  1066
Status            1200      2    Placed   716
```

Based on cardinality, it does not appear that these variables are misclassified.

2.1.2 Continuous Features

[11]:
```
jq.dqr_cont(df)
```

```
The non-categorical features are:
['Student ID', 'Undergrad Grade', 'MBA Grade', 'Employability (Before)',
'Employability (After)', 'Annual Salary']
Data Quality for Continous Features
Total Features: 6 / 1200 rows
```

	Feature	Count	Missing	% missing	Cardinality
0	Student ID	1200	0	0.00	1200
1	Undergrad Grade	1164	36	3.00	762
2	MBA Grade	1200	0	0.00	837
3	Employability (Before)	1193	7	0.58	1182
4	Employability (After)	1195	5	0.42	776
5	Annual Salary	716	484	40.33	89

```
Descriptive Stats
```

	count	mean	std	min	25%	\
Student ID	1200.0	600.50	346.55	1.0	300.75	
Undergrad Grade	1164.0	56.56	22.31	10.0	39.88	
MBA Grade	1200.0	52.83	23.49	0.0	35.72	
Employability (Before)	1193.0	216.31	36.28	62.0	189.96	
Employability (After)	1195.0	288.08	124.53	62.0	197.86	
Annual Salary	716.0	125285.71	49343.71	20000.0	100500.00	

	50%	75%	max
Student ID	600.50	900.25	1200.00
Undergrad Grade	56.91	75.02	100.00
MBA Grade	53.20	72.47	96.10
Employability (Before)	215.24	239.66	423.00
Employability (After)	260.93	349.00	697.39
Annual Salary	100500.00	148000.00	470333.33

[12]:
```
# Verify that all 484 missing values in Annual Salary is due to Status == Not␣
 ↪Placed
nan_annual_salary = df[
    (df.Status == "Not Placed") & (df["Annual Salary"].isna())
].shape[0]
print(
    f"Number of NaN values in Annual Salary where Status == 'Not Placed':␣
 ↪{nan_annual_salary}."
)
```

```
Number of NaN values in Annual Salary where Status == 'Not Placed': 484.
```

These four columns had missing values. Skewness of the distribution will determine if imputation

will be done with the mean or median.

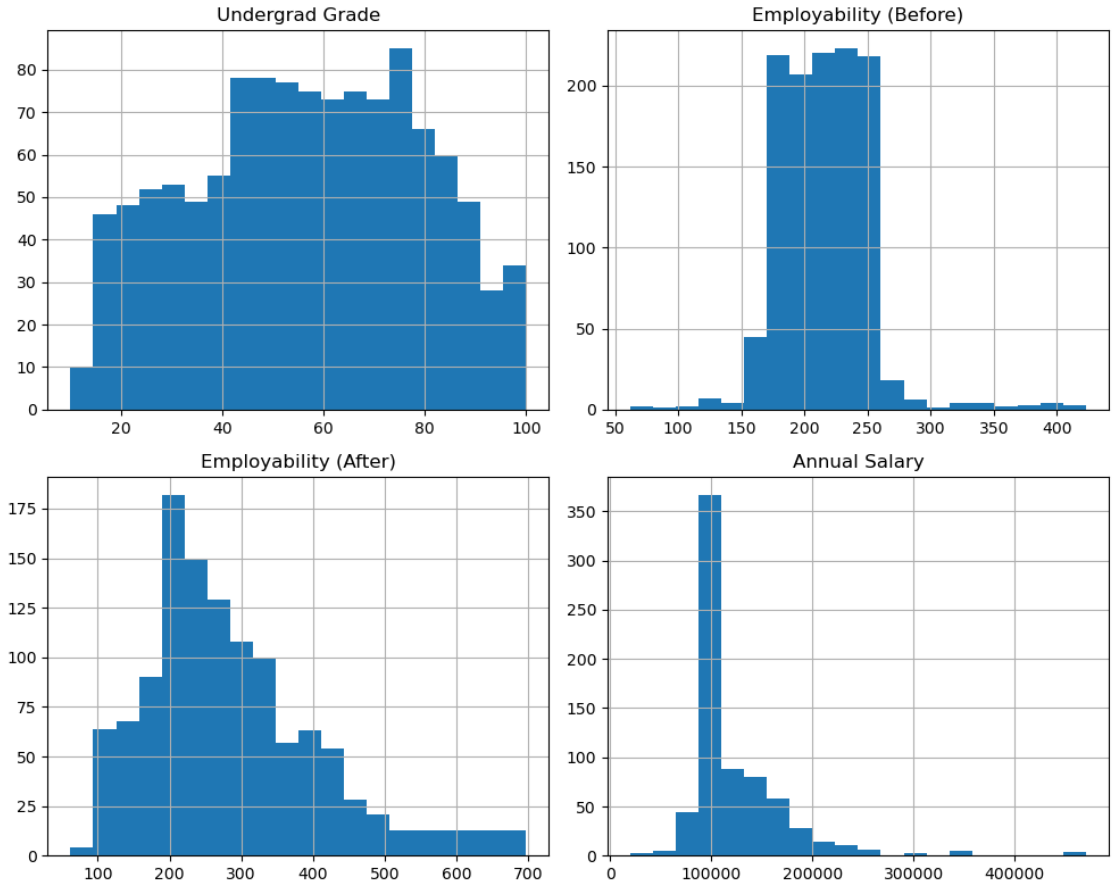
The missing values in the Annual Salary column were due to students not having a job after graduation. Therefore, these will be imputed with "0".

2.2 Clean missing values

[13]:
```
cols = df.columns.tolist()
cols
```

[13]:
```
['Student ID',
 'Undergrad Degree',
 'Undergrad Grade',
 'MBA Grade',
 'Work Experience',
 'Employability (Before)',
 'Employability (After)',
 'Status',
 'Annual Salary']
```

[14]:
```
missing_cols = [
    "Undergrad Grade",
    "Employability (Before)",
    "Employability (After)",
    "Annual Salary",
]
df[missing_cols].hist(bins=20, figsize=(10, 8))
plt.tight_layout()
plt.show()
```



Based on the skewness of the distribution shown in the charts, the median will be used to impute missing values.

Missing values in Annual Salary will be imputed with 0.

2.2.1 Impute Missing Values

[15]:
```
df["Undergrad Grade"].fillna(df["Undergrad Grade"].median(), inplace=True)
df["Employability (Before)"].fillna(df["Employability (Before)"].median(),␣
 ↪inplace=True)
df["Employability (After)"].fillna(df["Employability (After)"].median(),␣
 ↪inplace=True)
# Replace missing values with 0 for Annual Salary.
df["Annual Salary"].fillna(0, inplace=True)
```

[16]:
```
jq.dqr_cont(df)
```

```
The non-categorical features are:
['Student ID', 'Undergrad Grade', 'MBA Grade', 'Employability (Before)',
'Employability (After)', 'Annual Salary']
```

```
Data Quality for Continous Features
Total Features: 6 / 1200 rows
```

	Feature	Count	Missing	% missing	Cardinality
0	Student ID	1200	0	0.0	1200
1	Undergrad Grade	1200	0	0.0	762
2	MBA Grade	1200	0	0.0	837
3	Employability (Before)	1200	0	0.0	1181
4	Employability (After)	1200	0	0.0	775
5	Annual Salary	1200	0	0.0	89

```
Descriptive Stats
```

	count	mean	std	min	25%	50%	\
Student ID	1200.0	600.50	346.55	1.0	300.75	600.50	
Undergrad Grade	1200.0	56.57	21.97	10.0	40.50	56.91	
MBA Grade	1200.0	52.83	23.49	0.0	35.72	53.20	
Employability (Before)	1200.0	216.30	36.17	62.0	190.11	215.24	
Employability (After)	1200.0	287.97	124.28	62.0	197.96	260.93	
Annual Salary	1200.0	74753.81	72336.36	0.0	0.00	100500.00	

	75%	max
Student ID	900.25	1200.00
Undergrad Grade	74.53	100.00
MBA Grade	72.47	96.10
Employability (Before)	239.52	423.00
Employability (After)	348.40	697.39
Annual Salary	109000.00	470333.33

Verified that dataset no longer has any missing values.

3 Export Clean Dataset

[17]:
```
# Rename columns to remove spaces and brackets
df.columns = df.columns.str.replace(" ", "_").str.replace("(", "").str.
  ↪replace(")", "")
print(df.columns)
```

```
Index(['Student_ID', 'Undergrad_Degree', 'Undergrad_Grade', 'MBA_Grade',
       'Work_Experience', 'Employability_Before', 'Employability_After',
       'Status', 'Annual_Salary'],
      dtype='object')
```

[18]:
```
df_clean = df.copy()
df_clean.to_csv(datasets / "school_clean.csv", index=False)
```

[19]:
```
# Verify dataset was exported properly
dfc = pd.read_csv(datasets / "school_clean.csv")
dfc.head()
```

[19]:

	Student_ID	Undergrad_Degree	Undergrad_Grade	MBA_Grade	Work_Experience	\
0	1	Business	68.4	90.2	No	
1	2	Business	62.1	92.8	No	
2	3	Computer Science	70.2	68.7	Yes	
3	4	Engineering	75.1	80.7	No	
4	5	Finance	60.9	74.9	No	

	Employability_Before	Employability_After	Status	Annual_Salary
0	252.0	276.0	Placed	111000.0
1	423.0	410.0	Not Placed	0.0
2	101.0	119.0	Placed	107000.0
3	288.0	334.0	Not Placed	0.0
4	248.0	252.0	Not Placed	0.0

[ ]: