2187778-Inferential_Analysis

01. Replace the NaN values with correct value. And justify why you have chosen the same.

[1]:
```
import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
```

[2]:
```
dataset = pd.read_csv ("Placement.csv")
```

[3]:
```
dataset
```

[3]:

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
0	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.0	Mkt&HR	58.80	Placed	270000.0
1	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed	200000.0
2	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.0	Mkt&Fin	57.80	Placed	250000.0
3	4	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No	66.0	Mkt&HR	59.43	Not Placed	NaN
4	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.8	Mkt&Fin	55.50	Placed	425000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210	211	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	No	91.0	Mkt&Fin	74.49	Placed	400000.0
211	212	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	No	74.0	Mkt&Fin	53.62	Placed	275000.0
212	213	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	Yes	59.0	Mkt&Fin	69.72	Placed	295000.0
213	214	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	No	70.0	Mkt&HR	60.23	Placed	204000.0
214	215	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	No	89.0	Mkt&HR	60.22	Not Placed	NaN

215 rows × 15 columns

[5]:
```
# Check null Values in dataset
dataset.isna().sum()
```

[5]:
```
sl_no             0
gender            0
ssc_p             0
ssc_b             0
hsc_p             0
hsc_b             0
hsc_s             0
degree_p          0
degree_t          0
workex            0
etest_p           0
specialisation    0
mba_p             0
status            0
salary           67
dtype: int64
```

[12]:
```
import warnings
warnings.filterwarnings("ignore")
dataset["salary"].fillna(0, inplace=True) # who have null Value they don't have work or not yet place.
```

[13]:
```
dataset.isna().sum()
```

[13]:
```
sl_no             0
gender            0
ssc_p             0
ssc_b             0
hsc_p             0
hsc_b             0
hsc_s             0
degree_p          0
degree_t          0
workex            0
etest_p           0
specialisation    0
mba_p             0
status            0
salary            0
dtype: int64
```

02. How many of them are not placed?

```
print("How many of them are not Placed? \n", dataset["status"].value_counts()["Not Placed"])
```

```
How many of them are not Placed?
 67
```

03. Find the reason for non-placement from the dataset?

[27]:
```
df_notplaced = dataset[dataset["status"]=="Not Placed"]
df_notplaced # reason for non-placement from the dataset
```

[27]:

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
3	4	M	56.0	Central	52.0	Central	Science	52.00	Sci&Tech	No	66.00	Mkt&HR	59.43	Not Placed	0.0
5	6	M	55.0	Others	49.8	Others	Science	67.25	Sci&Tech	Yes	55.00	Mkt&Fin	51.58	Not Placed	0.0
6	7	F	46.0	Others	49.2	Others	Commerce	79.00	Comm&Mgmt	No	74.28	Mkt&Fin	53.29	Not Placed	0.0
9	10	M	58.0	Central	70.0	Central	Commerce	61.00	Comm&Mgmt	No	54.00	Mkt&Fin	52.21	Not Placed	0.0
12	13	F	47.0	Central	55.0	Others	Science	65.00	Comm&Mgmt	No	62.00	Mkt&HR	65.04	Not Placed	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
198	199	F	67.0	Central	70.0	Central	Commerce	65.00	Others	No	88.00	Mkt&HR	71.96	Not Placed	0.0
201	202	M	54.2	Central	63.0	Others	Science	58.00	Comm&Mgmt	No	79.00	Mkt&HR	58.44	Not Placed	0.0
206	207	M	41.0	Central	42.0	Central	Science	60.00	Comm&Mgmt	No	97.00	Mkt&Fin	53.39	Not Placed	0.0
208	209	F	43.0	Central	60.0	Others	Science	65.00	Comm&Mgmt	No	92.66	Mkt&HR	62.92	Not Placed	0.0
214	215	M	62.0	Central	58.0	Others	Science	53.00	Comm&Mgmt	No	89.00	Mkt&HR	60.22	Not Placed	0.0

67 rows × 15 columns

```
dataset.dtypes
```

```
sl_no              int64
gender            object
ssc_p            float64
ssc_b             object
hsc_p            float64
hsc_b             object
hsc_s             object
degree_p         float64
degree_t          object
workex            object
etest_p          float64
specialisation    object
mba_p            float64
status            object
salary           float64
dtype: object
```

```
def QuanQual_np(dataset):
    quan_np = []
    qual_np = []
    for columnName in dataset.columns:
        #print(columnName)
        if (dataset[columnName].dtype=='O'):
            #print ("Qual")
            qual_np.append (columnName)
        else:
            #print("quan")
            quan_np.append (columnName)
    return quan_np,qual_np
```

```
descriptive=pd.DataFrame(index=["Median"],columns=quan_np)
for columnName in quan_np:
    descriptive[columnName]["Median"]=df_notplaced[columnName].median()
```

```
df_placed = dataset[dataset["status"]=="Placed"]
df_placed
```

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
0	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.0	Mkt&HR	58.80	Placed	270000.0
1	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed	200000.0
2	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.0	Mkt&Fin	57.80	Placed	250000.0
4	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.8	Mkt&Fin	55.50	Placed	425000.0
7	8	M	82.00	Central	64.00	Central	Science	66.00	Sci&Tech	Yes	67.0	Mkt&Fin	62.14	Placed	252000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
209	210	M	62.00	Central	72.00	Central	Commerce	65.00	Comm&Mgmt	No	67.0	Mkt&Fin	56.49	Placed	216000.0
210	211	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	No	91.0	Mkt&Fin	74.49	Placed	400000.0
211	212	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	No	74.0	Mkt&Fin	53.62	Placed	275000.0
212	213	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	Yes	59.0	Mkt&Fin	69.72	Placed	295000.0
213	214	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	No	70.0	Mkt&HR	60.23	Placed	204000.0

```
def QuanQual_pl(dataset):
    quan_pl = []
    qual_pl = []
    for columnName in dataset.columns:
        #print(columnName)
        if (dataset[columnName].dtype=='O'):
            #print ("Qual")
            qual_pl.append (columnName)
        else:
            #print("quan")
            quan_pl.append (columnName)
    return quan_pl,qual_pl
```

```
quan_pl,qual_pl = QuanQual_pl(dataset)
```

```
quan_pl
```

[32]:
```
descriptive1=pd.DataFrame(index=["Median"],columns=quan_pl)
for columnName in quan_pl:
    descriptive1[columnName]["Median"]=df_placed[columnName].median()
```

[35]:
```
print("Not Placed Median: \n",descriptive)
print ("Placed Median: \n",descriptive1)
```

```
Not Placed Median:
         sl_no  ssc_p  hsc_p degree_p etest_p  mba_p salary
Median  107.0  56.28  60.33     61.0    67.0  60.69    0.0
Placed Median:
         sl_no ssc_p hsc_p degree_p etest_p   mba_p    salary
Median  108.5  72.5  68.0     68.0    72.0  62.245  265000.0
```

[ ]:
```
# Based on comparation between Not Place and Placed status. Who got etest_ below 70% they are not placed and who got etest_p above 70% they are placed.
```

```
notplaced=pd.DataFrame(descriptive)
placed = pd.DataFrame(descriptive1)
```

```
from tabulate import tabulate
print ("\nNOT PLACED")
print(tabulate(notplaced, headers = 'keys', tablefmt = 'psql'))
print ("\nPLACED")
print(tabulate(placed, headers = 'keys', tablefmt = 'psql'))
```

NOT PLACED

	sl_no	ssc_p	hsc_p	degree_p	etest_p	mba_p	salary
Median	107	56.28	60.33	61	67	60.69	0

PLACED

	sl_no	ssc_p	hsc_p	degree_p	etest_p	mba_p	salary
Median	108.5	72.5	68	68	72	62.245	265000

- **Based on comparation between Not Place and Placed status. Who got etest_ below 70% they are not placed and who got etest_p above 70% they are placed.**
- **Compare with placed applicants, who do not place applicants' low performance in school and higher studies.**

04. What kind of relation between salary and mba_p?

[29]:
```
dataset_corr = dataset.select_dtypes(include=['number']).corr()
dataset_corr
```

[29]:

	sl_no	ssc_p	hsc_p	degree_p	etest_p	mba_p	salary
sl_no	1.000000	-0.078155	-0.085711	-0.088281	0.063636	0.022327	0.002543
ssc_p	-0.078155	1.000000	0.511472	0.538404	0.261993	0.388478	0.538090
hsc_p	-0.085711	0.511472	1.000000	0.434206	0.245113	0.354823	0.452569
degree_p	-0.088281	0.538404	0.434206	1.000000	0.224470	0.402364	0.408371
etest_p	0.063636	0.261993	0.245113	0.224470	1.000000	0.218055	0.186988
mba_p	0.022327	0.388478	0.354823	0.402364	0.218055	1.000000	0.139823
salary	0.002543	0.538090	0.452569	0.408371	0.186988	0.139823	1.000000

[30]:
```
corr_salary_mba = dataset['salary'].corr(dataset['mba_p'])
print(f"\nCorrelation between salary and mba_p: {corr_salary_mba * 100:.0f}%")
```

```
Correlation between salary and mba_p: 14%
```

- **Correlation between salary, and mba_p. nearly 14% Directly proportional. It is a Positive Correlation.**

05. Which specialization is getting minimum salary?

```
min_spec = dataset.groupby('specialisation')['salary'].median().sort_values().head(1)
print("\nSpecialisation with minimum median salary:\n", min_spec)
```

```
Specialisation with minimum median salary:
 specialisation
Mkt&HR    210000.0
Name: salary, dtype: float64
```

- **Mk&HR is getting minimum salary in this dataset**

06. How many of them are getting above 500000 salaries?

[44]:
```
above_500k= dataset[dataset['salary'] > 500000]

print("Details of students with salary above 500000:")
above_500k
```

```
Details of students with salary above 500000:
```

[44]:

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
119	120	M	60.8	Central	68.40	Central	Commerce	64.6	Comm&Mgmt	Yes	82.66	Mkt&Fin	64.34	Placed	940000.0
150	151	M	71.0	Central	58.66	Central	Science	58.0	Sci&Tech	Yes	56.00	Mkt&Fin	61.30	Placed	690000.0
177	178	F	73.0	Central	97.00	Others	Commerce	79.0	Comm&Mgmt	Yes	89.00	Mkt&Fin	70.81	Placed	650000.0

07. Test the Analysis of Variance between etest_p and mba_p at significance level 5%.(Make decision using Hypothesis Testing)?

[44]:
```
from scipy.stats import f_oneway

f_stat, p_value = f_oneway(dataset['etest_p'], dataset['mba_p'])

print("F-statistic:", f_stat)
print("P-value:", p_value)

a = 0.05

if p_value < a:
    print("Accept the null hypothesis")
else:
    print("Rreject the null hypothesis:")
```

```
F-statistic: 98.64487057324708
P-value: 4.672547689133573e-21
Accept the null hypothesis
```

08. Test the similarity between the degree_t(Sci&Tech) and specialization (Mkt&HR) with respect to salary at significance level of 5%. (Make decision using Hypothesis Testing)

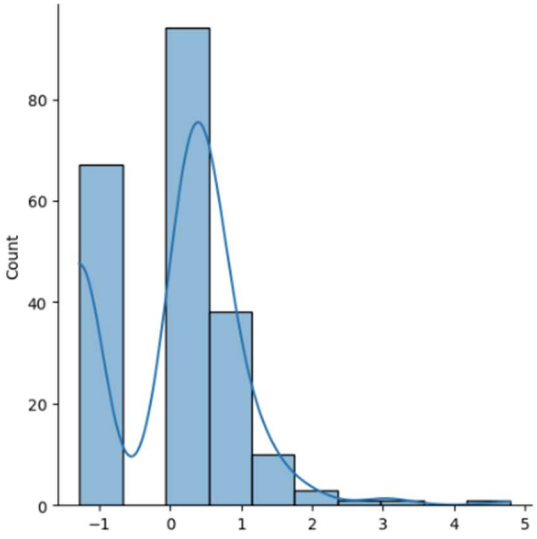
Method - T-test. → Independent Sample-Unpaired Different group(degree_t, spcialization) but same condition (salary).

[46]:
```
from scipy.stats import ttest_ind
dataset=dataset.dropna()
group_degree = dataset[dataset["degree_t"]=="Sci&Tech"]["salary"]
group_spec = dataset[dataset["specialisation"]=="Mkt&HR"]["salary"]

t_stat2, p_val2 = stats.ttest_ind(group_degree, group_spec)# equal_var=False)
print("\nT-test (Sci&Tech vs Mkt&HR) salaries:")
print("t-statistic:", t_stat2, ", p-value:", p_val2)
if p_val2 < 0.05:
    print("Reject H0 → Salary means differ")
else:
    print("Fail to Reject H0 → Salary means similar")
```

```
T-test (Sci&Tech vs Mkt&HR) salaries:
t-statistic: 2.692041243555374 , p-value: 0.007897969943471179
Reject H0 → Salary means differ
```

09. Convert the normal distribution to the standard normal distribution for salary columns?

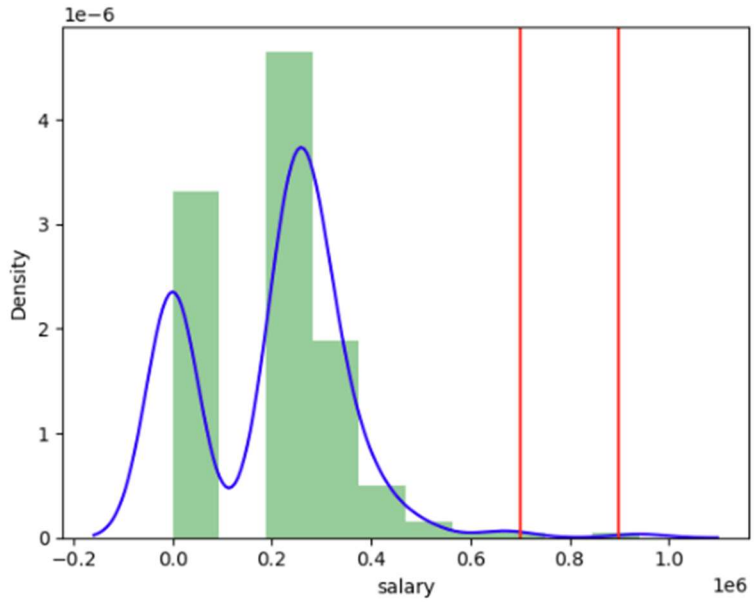
```
[68]: def stdNBgraph(dataset):
          # Converted to standard Normal Distribution
          import seaborn as sns
          mean = dataset.mean()
          std = dataset.std()
          values = [i for i in dataset]
          z_score = [((j-mean)/std) for j in values]
          sns.displot(z_score,kde=True)
          sum(z_score)/len(z_score)
          #z_score.std()

[71]: stdNBgraph(dataset['salary'])
```



10. What is the probability of Density Function of the salary range from 700000 to 900000?

```
[50]: def get_pdf_probability (dataset,startrange,endrange):
          from matplotlib import pyplot
          from scipy.stats import norm
          import seaborn as sns
          ax = sns.distplot(dataset,kde=True,kde_kws={'color':'blue'},color='Green')
          pyplot.axvline(startrange,color='Red')
          pyplot.axvline(endrange,color='Red')
          #generate a Sample
          sample = dataset
          #calculate parameters
          sample_mean = sample.mean()
          sample_std = sample.std()
          print ("Mean=%.3f, Standard Deviation=%.3f" % (sample_mean,sample_std))
          #difine the distrubution
          dist = norm(sample_mean, sample_std)
          #sample probabilities for a range of outcomes.
          values = [value for value in range (startrange, endrange)]
          probabilities = [dist.pdf(value) for value in values]
          prob=sum(probabilities)
          print ("The area between range ({},{}):{}".format(startrange,endrange,sum(probabilities)))
          return prob
```

```
]: get_pdf_probability(dataset["salary"],700000,900000)

Mean=198702.326, Standard Deviation=154780.927
The area between range (700000,900000):0.0005973310593974868

]: np.float64(0.0005973310593974868)
```



11. Test the similarity between the degree_t(Sci&Tech)with respect to etest_p and mba_p at significance level of 5%. (Make decision using Hypothesis Testing)?

```
[52]: from scipy.stats import ttest_rel
      dataset= dataset.dropna()
      s_deg = dataset[dataset["degree_t"]=="Sci&Tech"]["etest_p"]
      s_deg1 = dataset[dataset["degree_t"]=="Sci&Tech"]["mba_p"]
      #print Male
      ttest_rel(s_deg,s_deg1)

[52]: TtestResult(statistic=np.float64(5.0049844583693615), pvalue=np.float64(5.517920600505392e-06), df=np.int64(58))
```

12. Which parameter is highly correlated with salary?

```
[53]: dataset_cor = dataset.select_dtypes(include=['number']).corr()

[54]: dataset_cor
```

[54]:

	sl_no	ssc_p	hsc_p	degree_p	etest_p	mba_p	salary
sl_no	1.000000	-0.078155	-0.085711	-0.088281	0.063636	0.022327	0.002543
ssc_p	-0.078155	1.000000	0.511472	0.538404	0.261993	0.388478	0.538090
hsc_p	-0.085711	0.511472	1.000000	0.434206	0.245113	0.354823	0.452569
degree_p	-0.088281	0.538404	0.434206	1.000000	0.224470	0.402364	0.408371
etest_p	0.063636	0.261993	0.245113	0.224470	1.000000	0.218055	0.186988
mba_p	0.022327	0.388478	0.354823	0.402364	0.218055	1.000000	0.139823
salary	0.002543	0.538090	0.452569	0.408371	0.186988	0.139823	1.000000

13. plot any useful graph and explain it.?

```
[55]: import seaborn as sb
```

```
[56]: sb.pairplot(dataset)
```

```
[56]: <seaborn.axisgrid.PairGrid at 0x29eb7eaf4d0>
```