Questions

Which factor influenced a candidate in getting placed?

Does percentage matters for one to get placed?

Which degree specialization is much demanded by corporate?

Play with the data conducting all statistical tests.

In [4]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import chi2_contingency
import numpy as np
```

In [6]:
```
csv_file_path = "Placement_Data_Full_Class.csv"
df = pd.read_csv(csv_file_path)
```

In [7]:
```
import pandas_profiling as pp


profile = pp.ProfileReport( df, title="Campus Recruitment Profile", html={"style": {"full_width": True}
profile
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	15
Number of observations	215
Missing cells	67
Missing cells (%)	2.1%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	25.3 KiB
Average record size in memory	120.6 B

Variable types

Numeric	7
Categorical	7
Boolean	1

Alerts

`ssc_p` is highly overall correlated with `degree_p` and 1 other fields (degree_p, status)	High correlation
`hsc_p` is highly overall correlated with `status`	High correlation

Out[7]:

In [3]:
```
# Display the first few rows of the dataset
print("First few rows:")
print(df.head())
```

```
First few rows:
   sl_no gender  ssc_p    ssc_b  hsc_p    hsc_b     hsc_s  degree_p  \
0      1      M  67.00   Others  91.00   Others  Commerce     58.00
1      2      M  79.33  Central  78.33   Others   Science     77.48
2      3      M  65.00  Central  68.00  Central      Arts     64.00
3      4      M  56.00  Central  52.00  Central   Science     52.00
4      5      M  85.80  Central  73.60  Central  Commerce     73.30

    degree_t workex  etest_p specialisation  mba_p      status    salary
0   Sci&Tech     No     55.0         Mkt&HR  58.80      Placed  270000.0
1   Sci&Tech    Yes     86.5        Mkt&Fin  66.28      Placed  200000.0
2  Comm&Mgmt     No     75.0        Mkt&Fin  57.80      Placed  250000.0
3   Sci&Tech     No     66.0         Mkt&HR  59.43  Not Placed       NaN
4  Comm&Mgmt     No     96.8        Mkt&Fin  55.50      Placed  425000.0
```

In [4]:
```
# Get information about the dataset
print("\nDataset information:")
print(df.info())
```

```
Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   sl_no           215 non-null    int64
 1   gender          215 non-null    object
 2   ssc_p           215 non-null    float64
 3   ssc_b           215 non-null    object
 4   hsc_p           215 non-null    float64
 5   hsc_b           215 non-null    object
 6   hsc_s           215 non-null    object
 7   degree_p        215 non-null    float64
 8   degree_t        215 non-null    object
 9   workex          215 non-null    object
 10  etest_p         215 non-null    float64
 11  specialisation  215 non-null    object
 12  mba_p           215 non-null    float64
 13  status          215 non-null    object
 14  salary          148 non-null    float64
dtypes: float64(6), int64(1), object(8)
memory usage: 25.3+ KB
None
```

In [5]:
```
# Generate summary statistics for numerical columns
print("\nSummary statistics:")
print(df.describe())
```

```
Summary statistics:
            sl_no       ssc_p       hsc_p    degree_p     etest_p       mba_p  \
count  215.000000  215.000000  215.000000  215.000000  215.000000  215.000000
mean   108.000000   67.303395   66.333163   66.370186   72.100558   62.278186
std     62.209324   10.827205   10.897509    7.358743   13.275956    5.833385
min      1.000000   40.890000   37.000000   50.000000   50.000000   51.210000
25%     54.500000   60.600000   60.900000   61.000000   60.000000   57.945000
50%    108.000000   67.000000   65.000000   66.000000   71.000000   62.000000
75%    161.500000   75.700000   73.000000   72.000000   83.500000   66.255000
max    215.000000   89.400000   97.700000   91.000000   98.000000   77.890000

              salary
count     148.000000
mean   288655.405405
std     93457.452420
min    200000.000000
25%    240000.000000
50%    265000.000000
75%    300000.000000
max    940000.000000
```

In [6]:
```
# Get information about the dataset
print("\nDataset information:")
print(df.info())
```

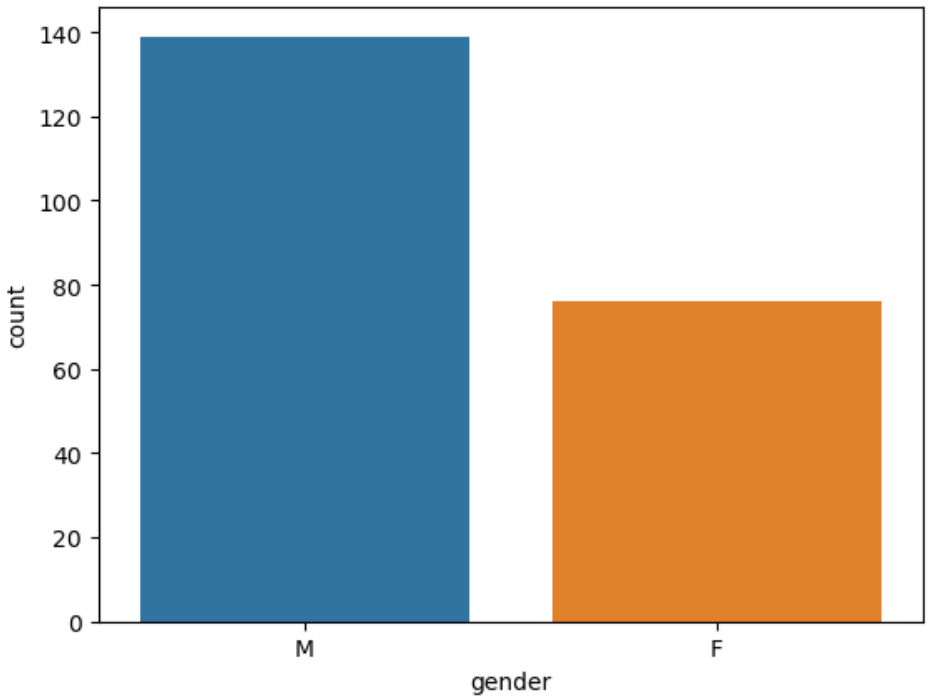
```
Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   sl_no           215 non-null    int64
 1   gender          215 non-null    object
 2   ssc_p           215 non-null    float64
 3   ssc_b           215 non-null    object
 4   hsc_p           215 non-null    float64
 5   hsc_b           215 non-null    object
 6   hsc_s           215 non-null    object
 7   degree_p        215 non-null    float64
 8   degree_t        215 non-null    object
 9   workex          215 non-null    object
 10  etest_p         215 non-null    float64
 11  specialisation  215 non-null    object
 12  mba_p           215 non-null    float64
 13  status          215 non-null    object
 14  salary          148 non-null    float64
dtypes: float64(6), int64(1), object(8)
memory usage: 25.3+ KB
None
```

Here we have tried to plot the features to get a basic idea of the data se

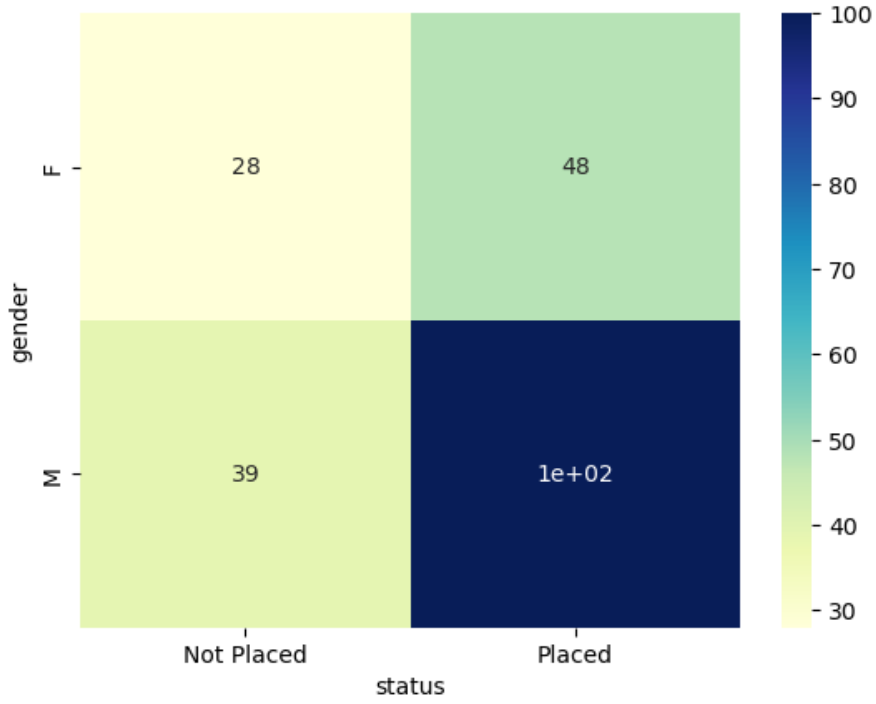
categorical with placed column

Gender

In [7]:
```
sns.countplot(x="gender", data=df)
plt.show()
```



In [8]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["gender"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```



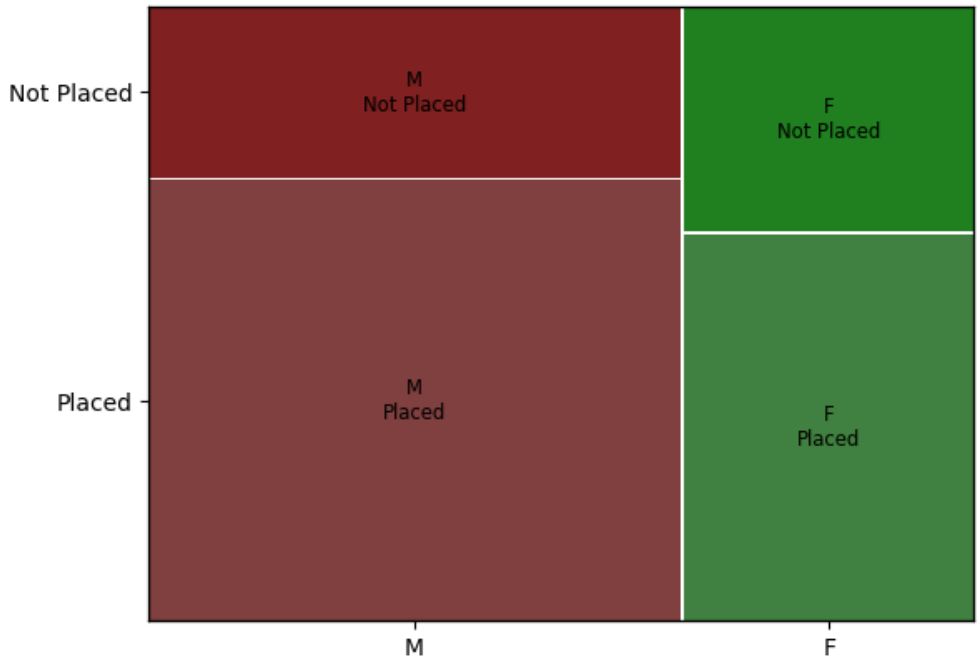
In [9]:
```
# Replace these with the actual column names
categorical_column1 = "gender"
categorical_column2 = "salary"
# Create a contingency table
contingency_table = pd.crosstab(df[categorical_column1], df[categorical_column2])
print(contingency_table)
```

```
salary  200000.0  204000.0  210000.0  216000.0  218000.0  220000.0  225000.0  \
gender
F              4         1         4         2         1         2         0
M              2         1         0         1         1         3         1

salary  230000.0  231000.0  233000.0  ...  393000.0  400000.0  411000.0  \
gender                                ...
F              2         0         0  ...         1         1         0
M              0         1         1  ...         0         3         1

salary  420000.0  425000.0  450000.0  500000.0  650000.0  690000.0  940000.0
gender
F              0         0         0         0         1         0         0
M              1         1         1         3         0         1         1

[2 rows x 45 columns]
```

In [10]:
```
# Perform the chi-square test
chi2, p, dof, expected = chi2_contingency(contingency_table)

print("Chi-Square:", chi2)
print("P-value:", p)
print("Degrees of Freedom:", dof)
```

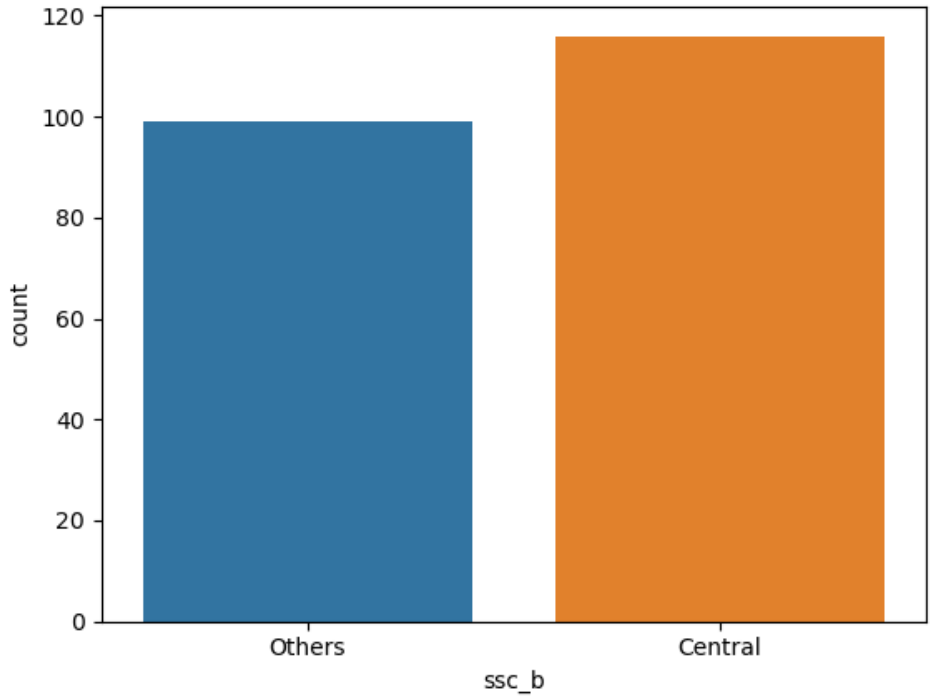
```
Chi-Square: 48.2760214045214
P-value: 0.3041748362314831
Degrees of Freedom: 44
```

In [11]:
```
from statsmodels.graphics.mosaicplot import mosaic

mosaic(df, ['gender', 'status'])
plt.show()
```

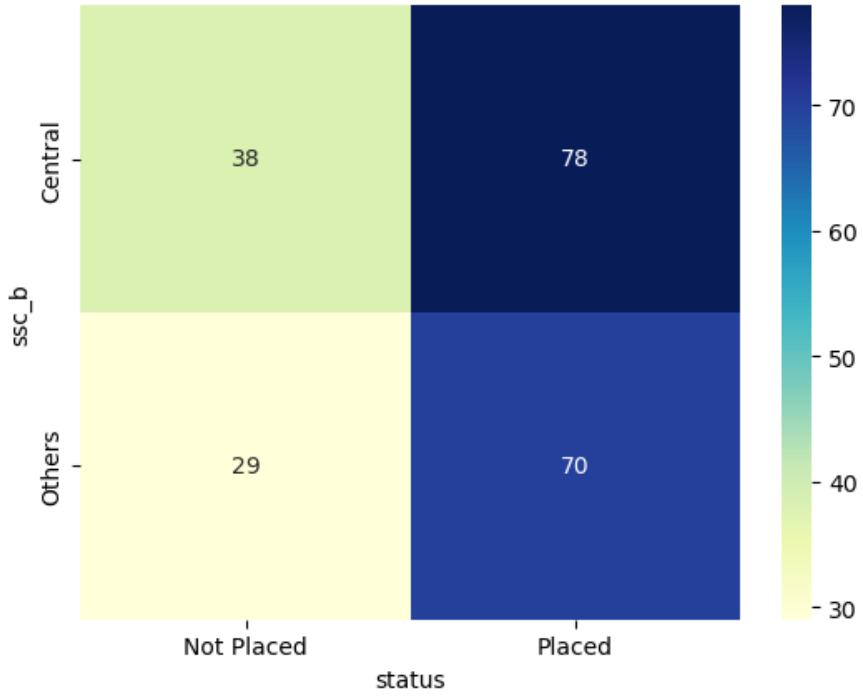


ssb

In [12]:
```
sns.countplot(x="ssc_b", data=df)
plt.show()
```



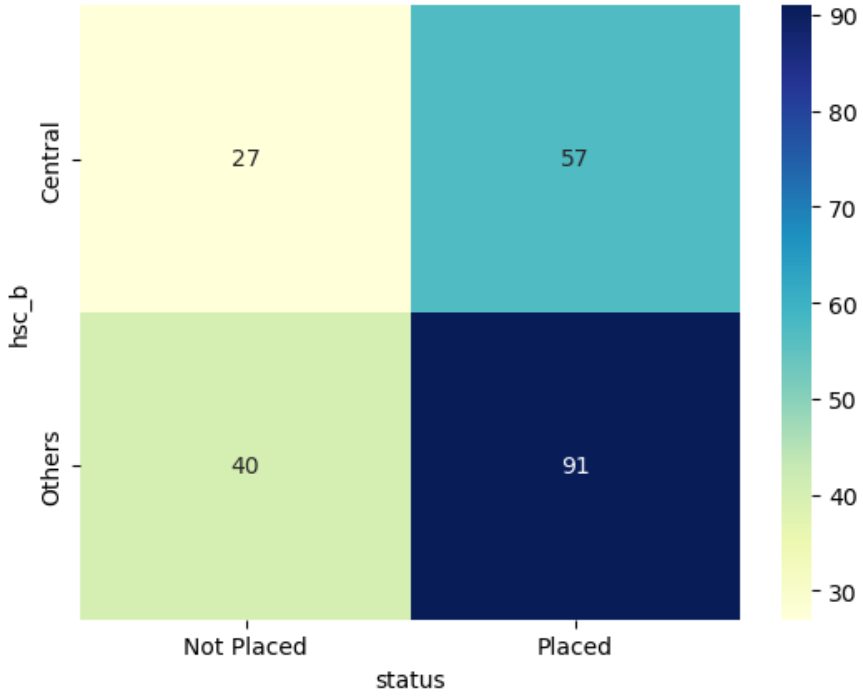
In [13]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["ssc_b"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```



hsc_b

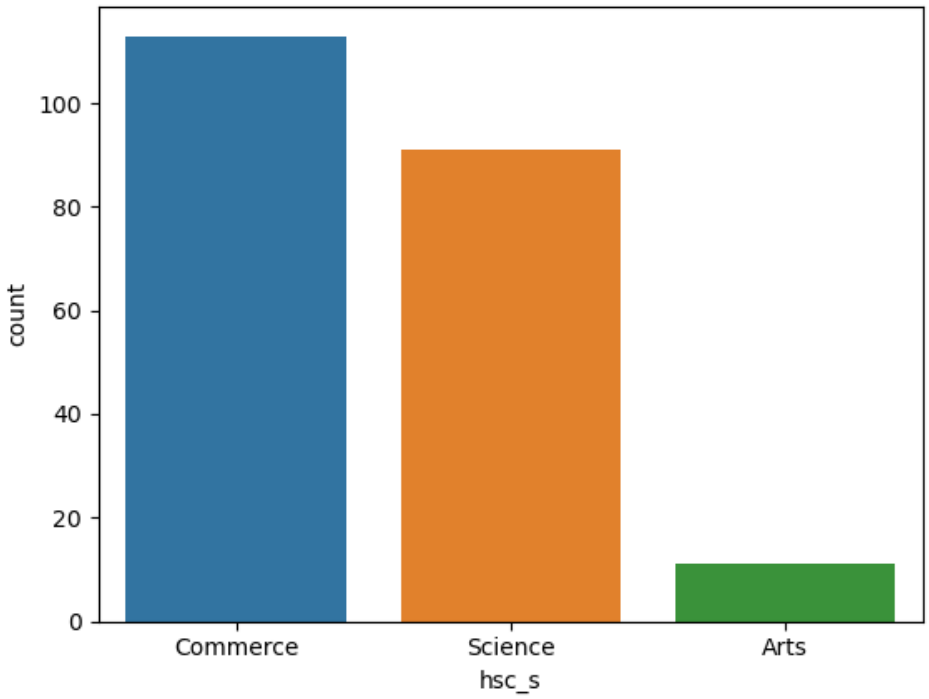
In [ ]:

In [14]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["hsc_b"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```

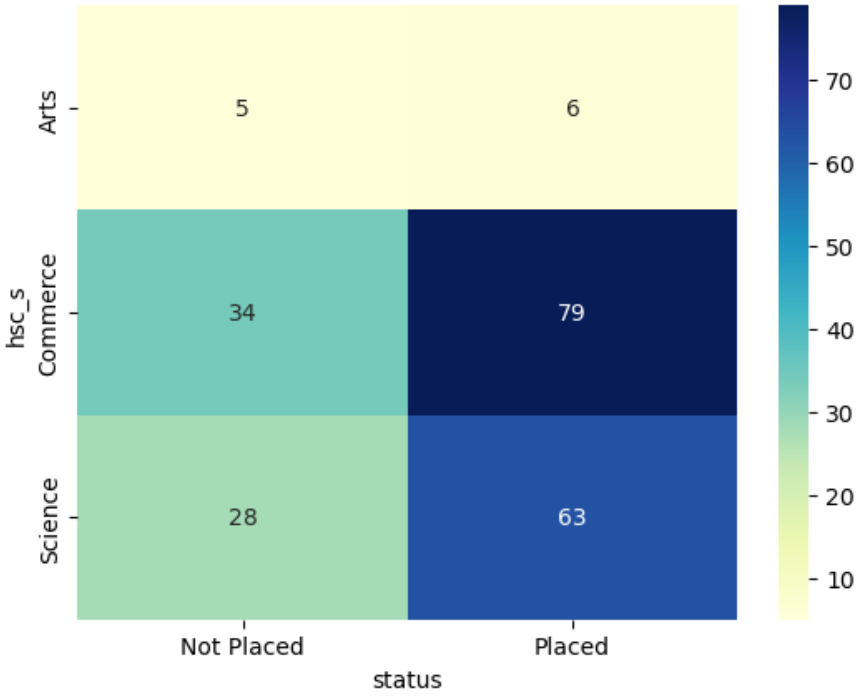


hsc_s

In [15]:
```
sns.countplot(x="hsc_s", data=df)
plt.show()
```

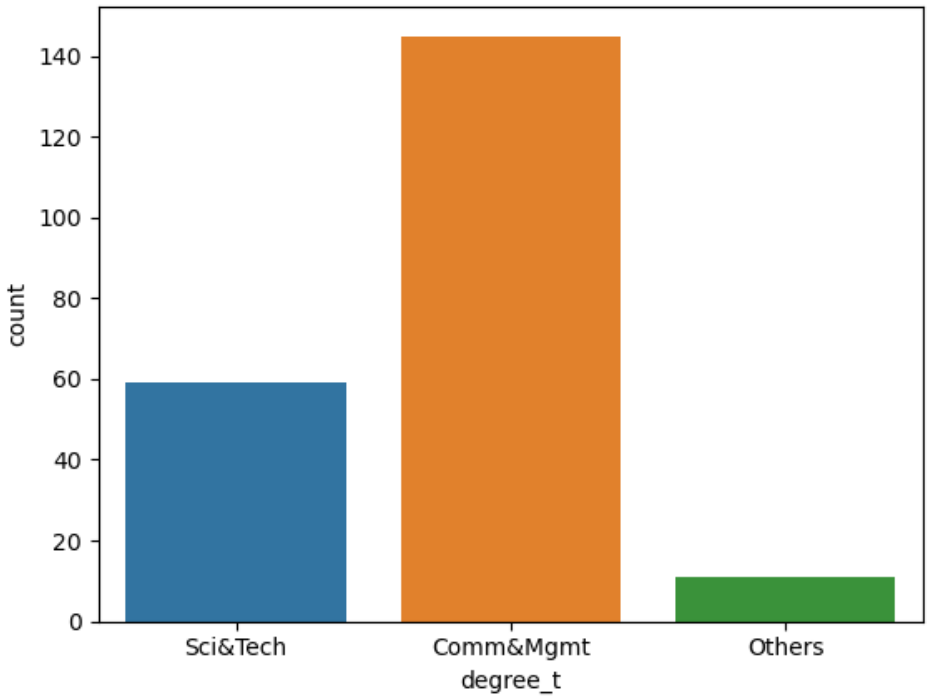


In [16]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["hsc_s"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```

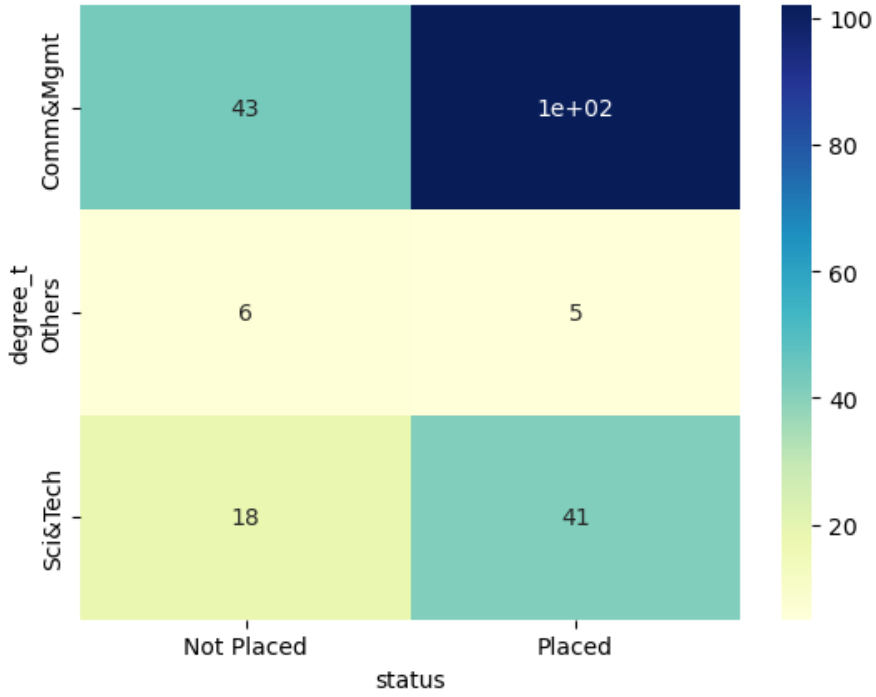


degree_t

In [17]:
```
sns.countplot(x="degree_t", data=df)
plt.show()
```

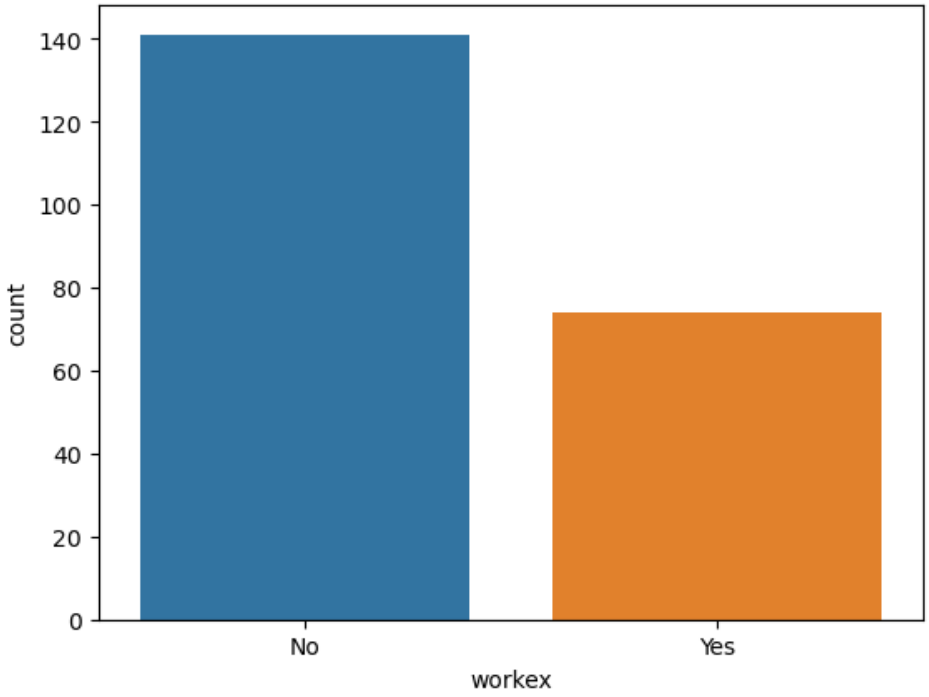


In [18]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["degree_t"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```

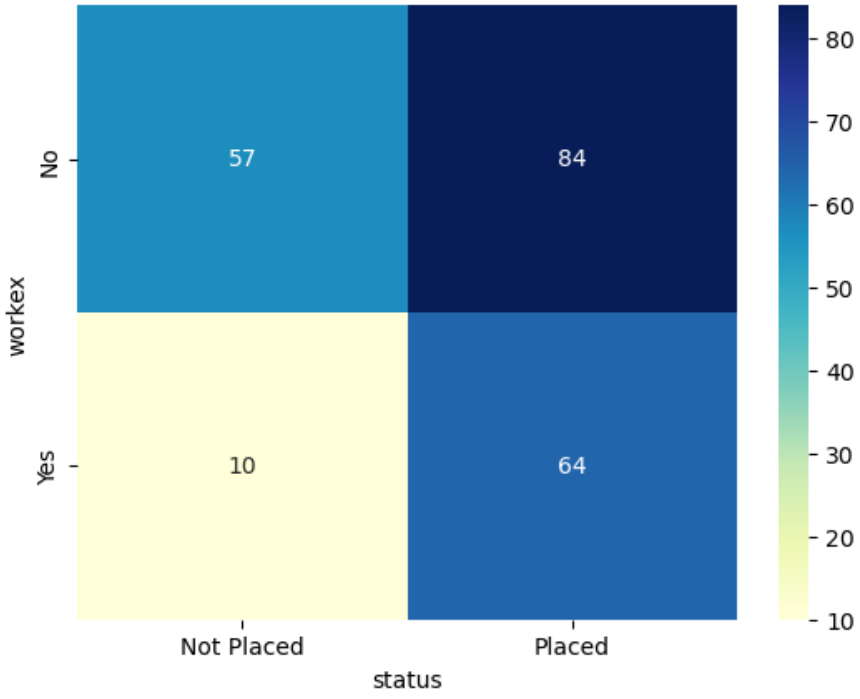


workex

In [19]:
```
sns.countplot(x="workex", data=df)
plt.show()
```

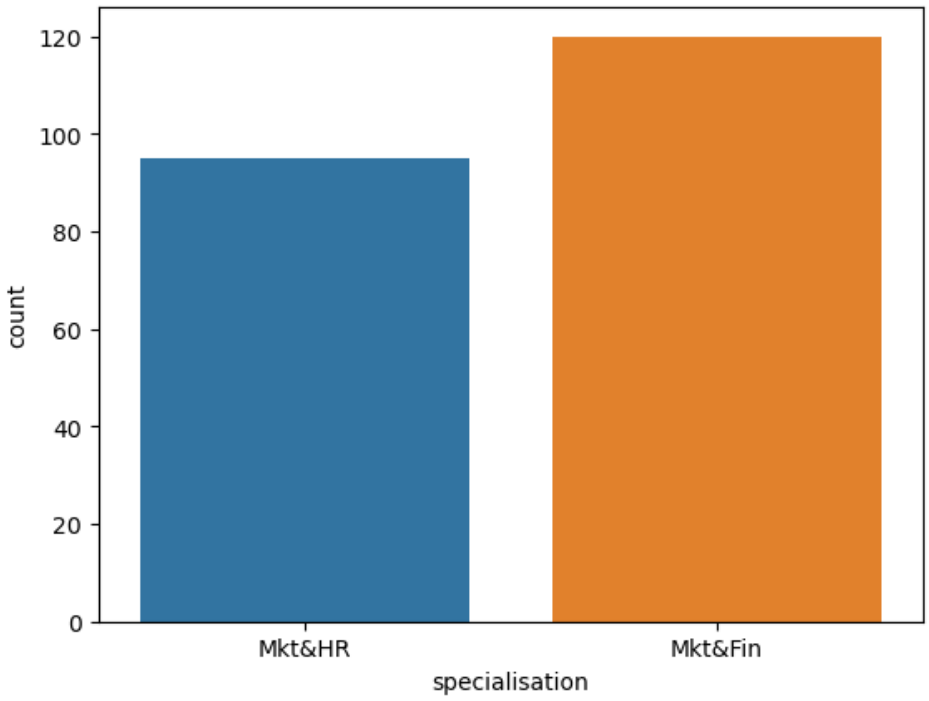


In [20]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["workex"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```

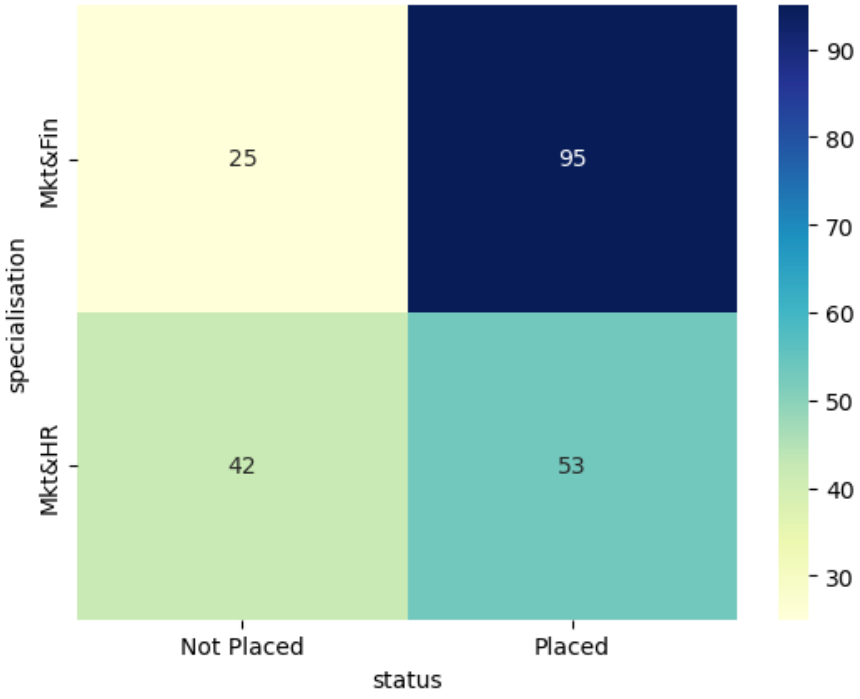


specialisation

In [21]:
```
sns.countplot(x="specialisation", data=df)
plt.show()
```



In [22]:
```
# Create a cross-tabulation of the two categorical variables
crosstab = pd.crosstab(df["specialisation"], df["status"])
# Plot the heatmap
sns.heatmap(crosstab, cmap="YlGnBu", annot=True)
plt.show()
```

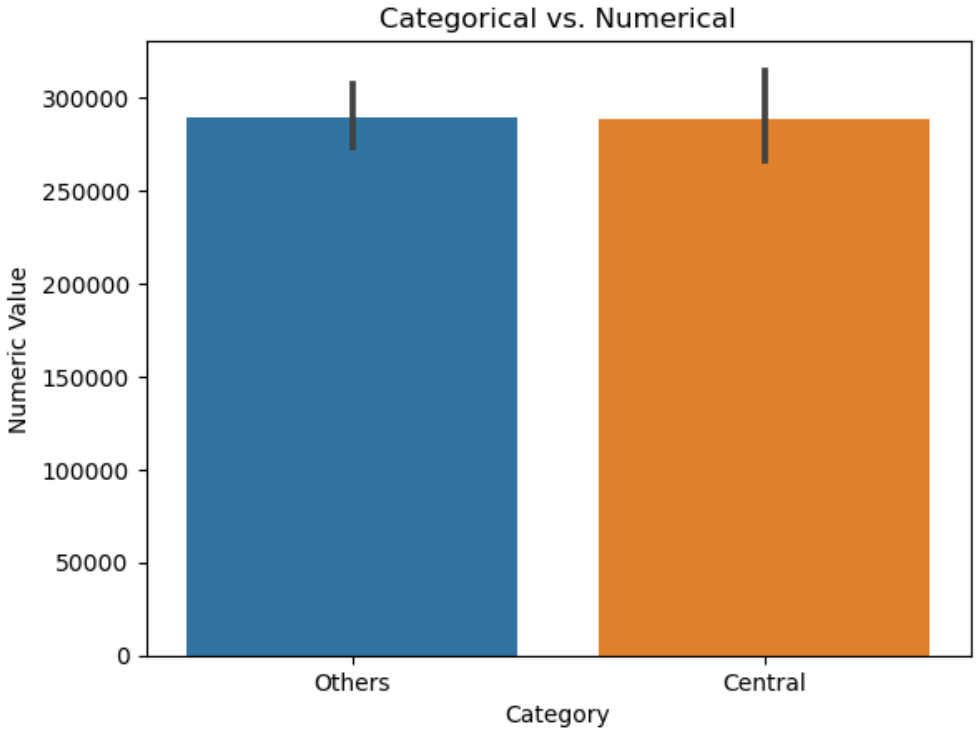


ssc_b with sallary

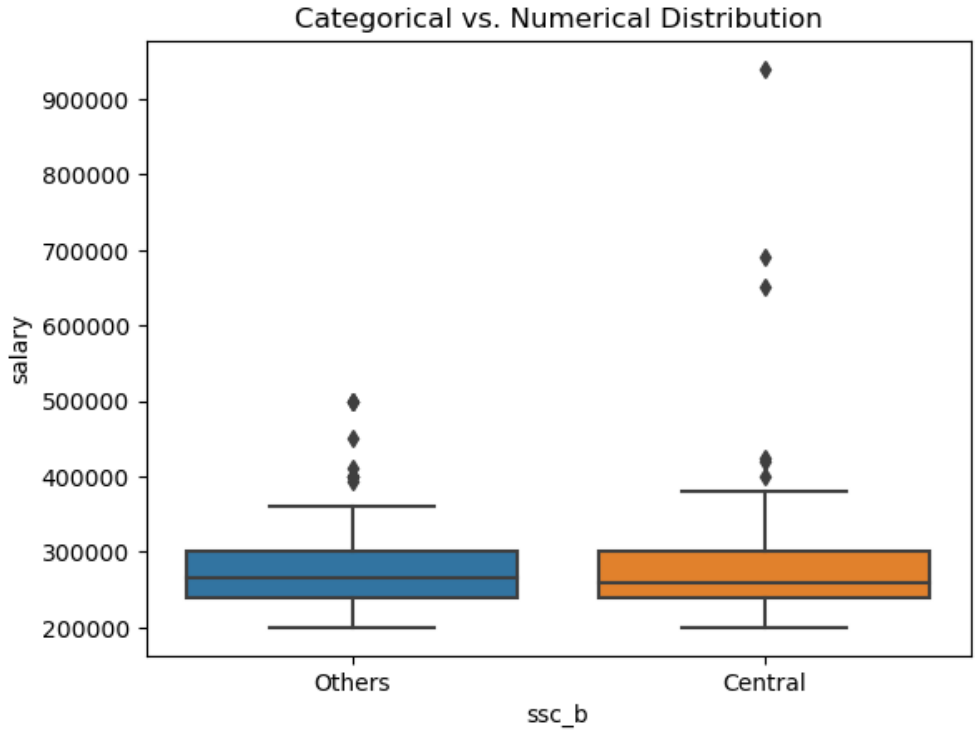
categorical with salary

In [23]:
```
import matplotlib.pyplot as plt
import seaborn as sns

sns.barplot(x='ssc_b', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
```

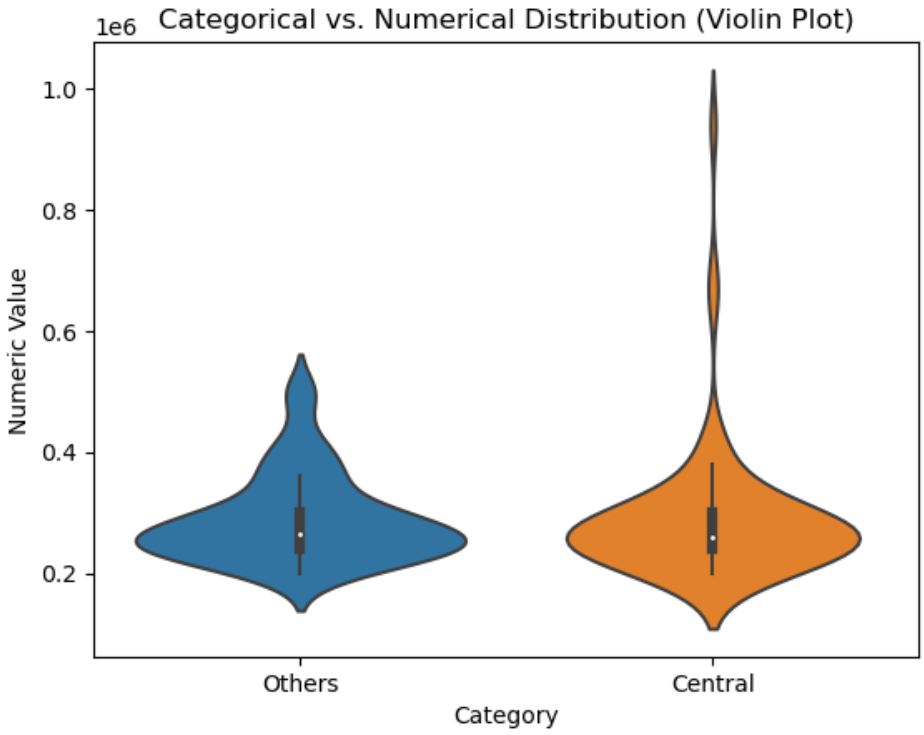
```
plt.title('Categorical vs. Numerical')
plt.show()
```



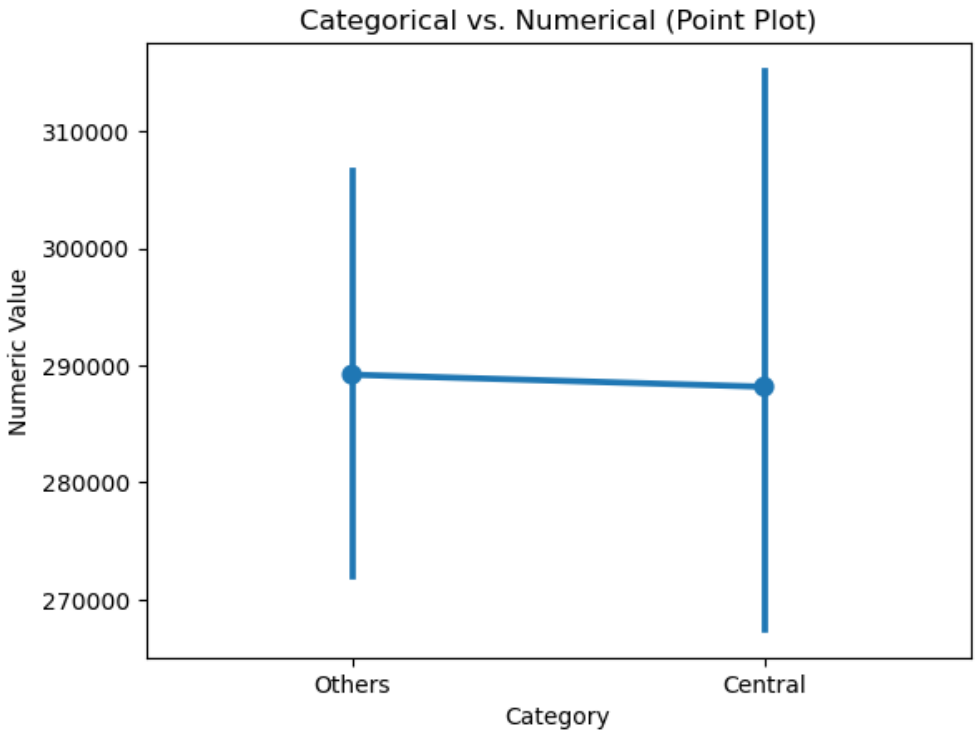
In [24]:
```
import matplotlib.pyplot as plt
import seaborn as sns

sns.boxplot(x='ssc_b', y='salary', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [25]:
```
sns.violinplot(x='ssc_b', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

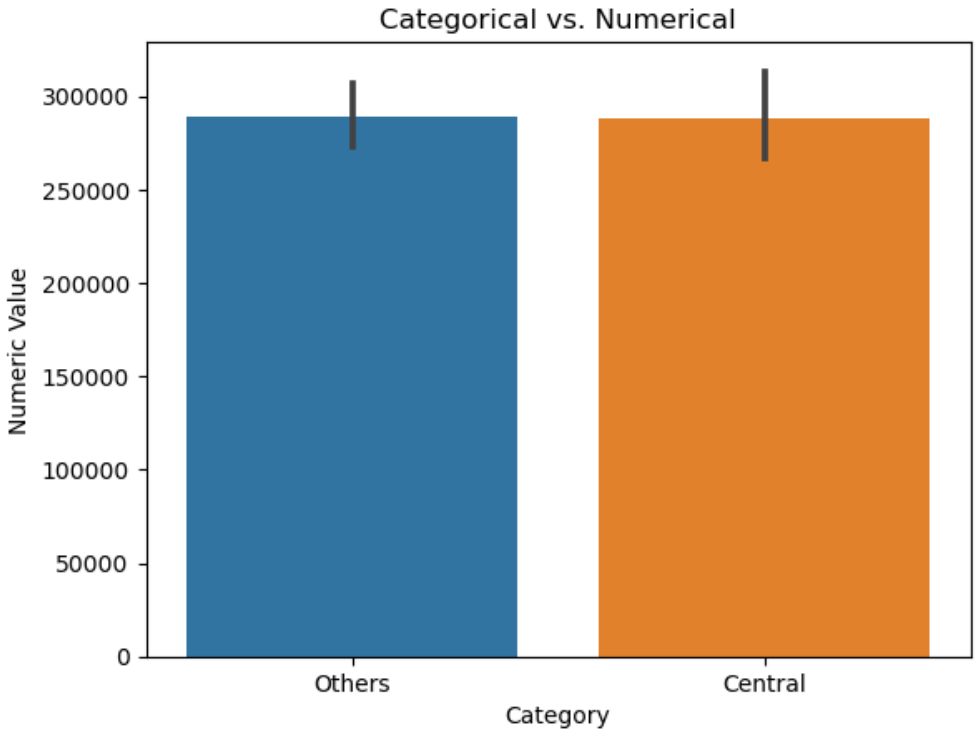


In [26]:
```
sns.pointplot(x='ssc_b', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

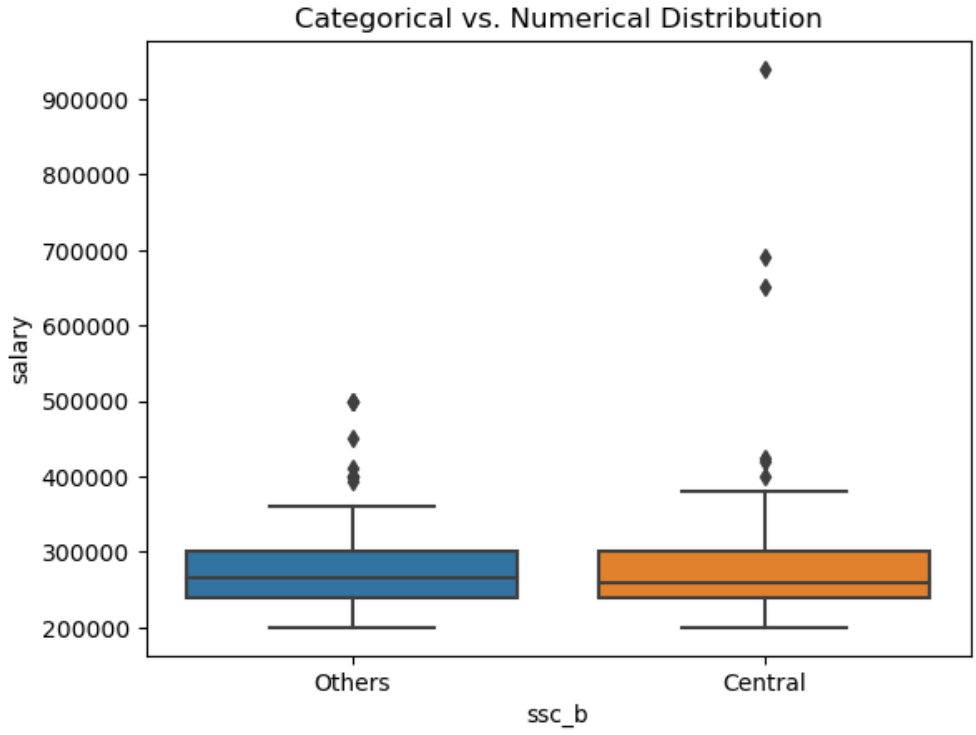


hsc_b

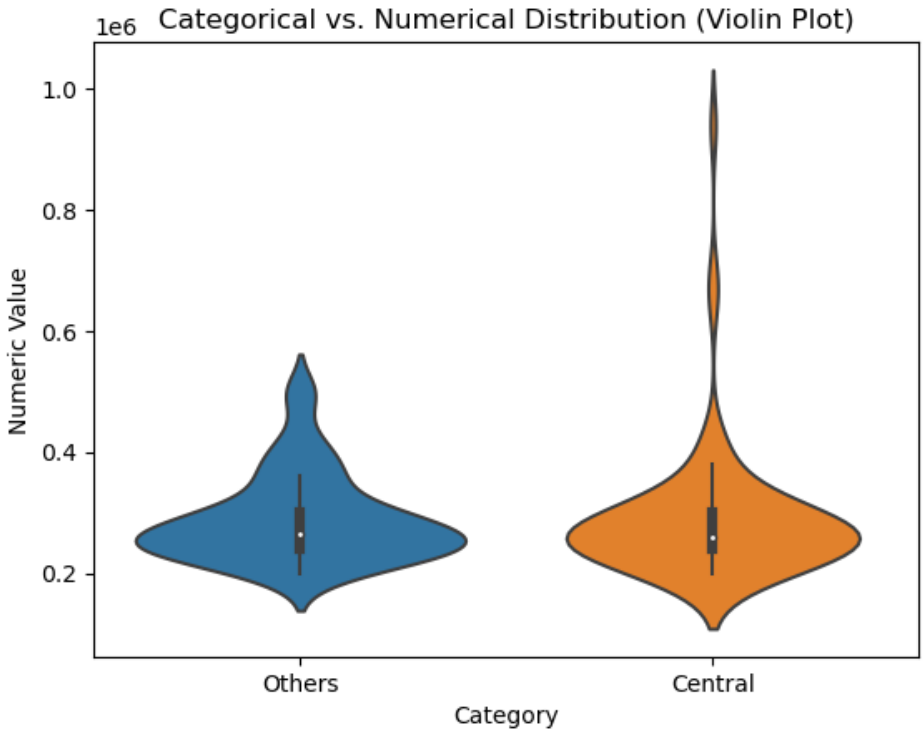
In [27]:
```
sns.barplot(x='ssc_b', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



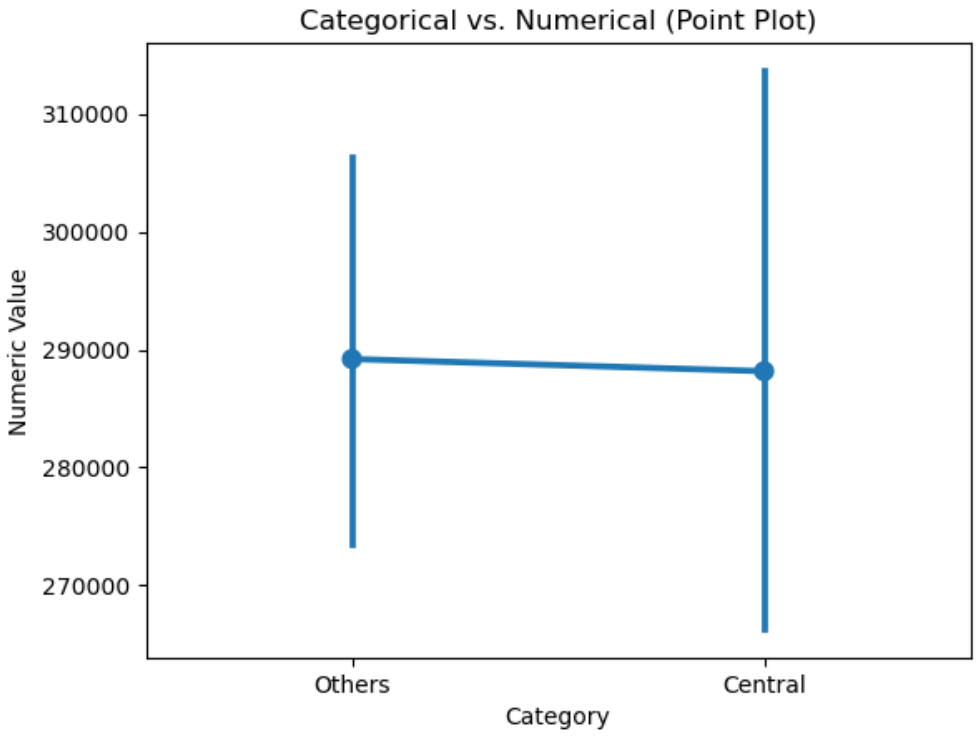
In [28]:
```
sns.boxplot(x='ssc_b', y='salary', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [29]:
```
sns.violinplot(x='ssc_b', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

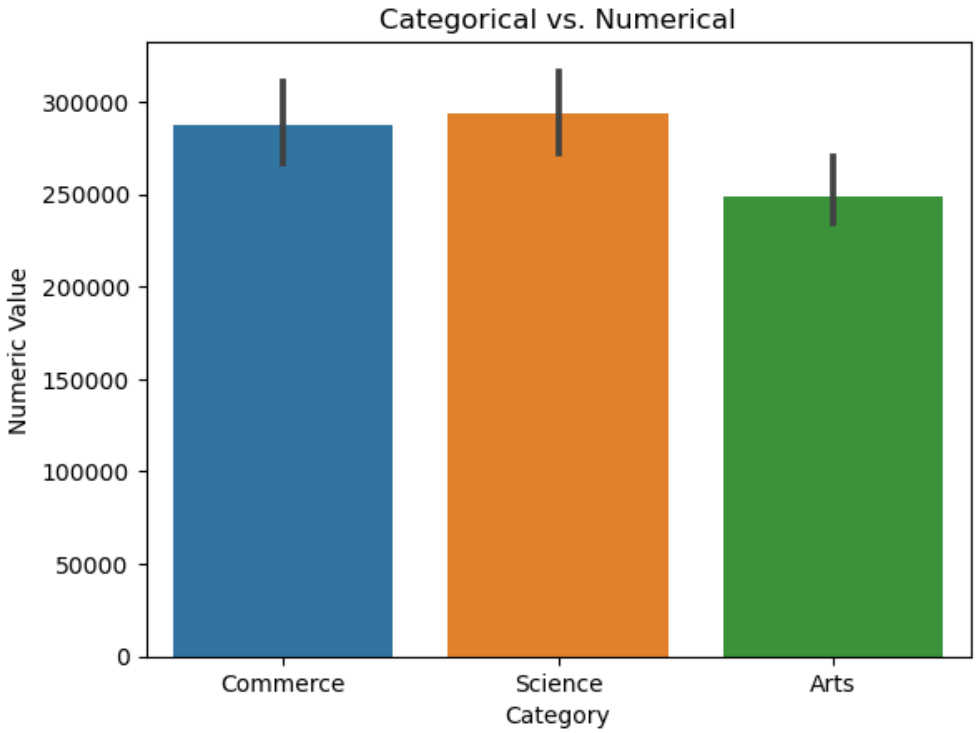


In [30]:
```
sns.pointplot(x='ssc_b', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

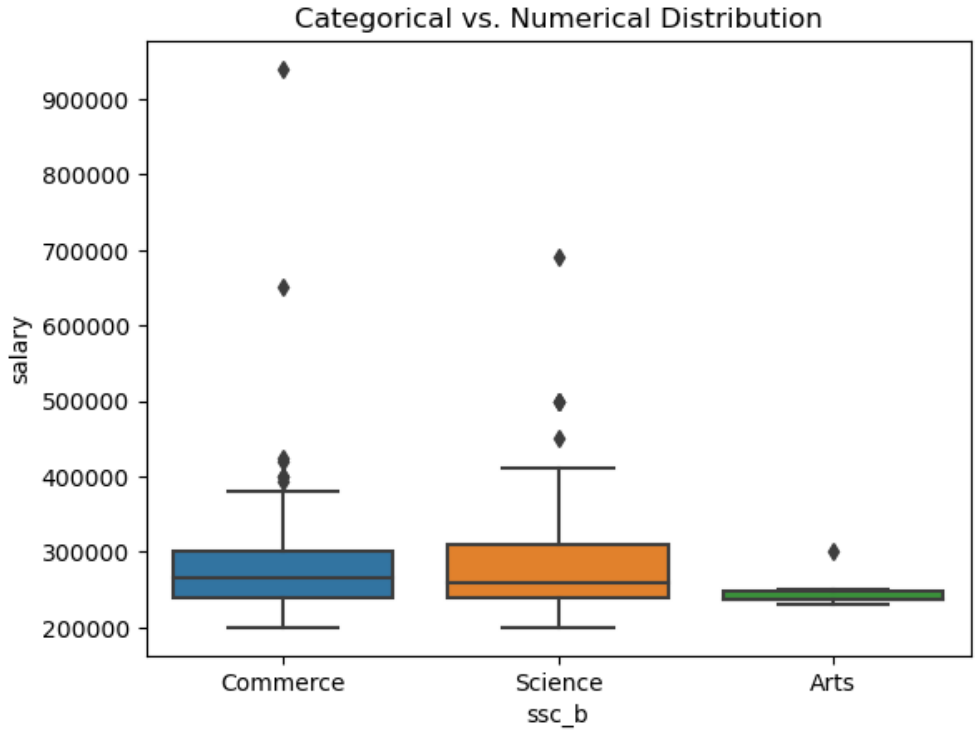


hsc_s

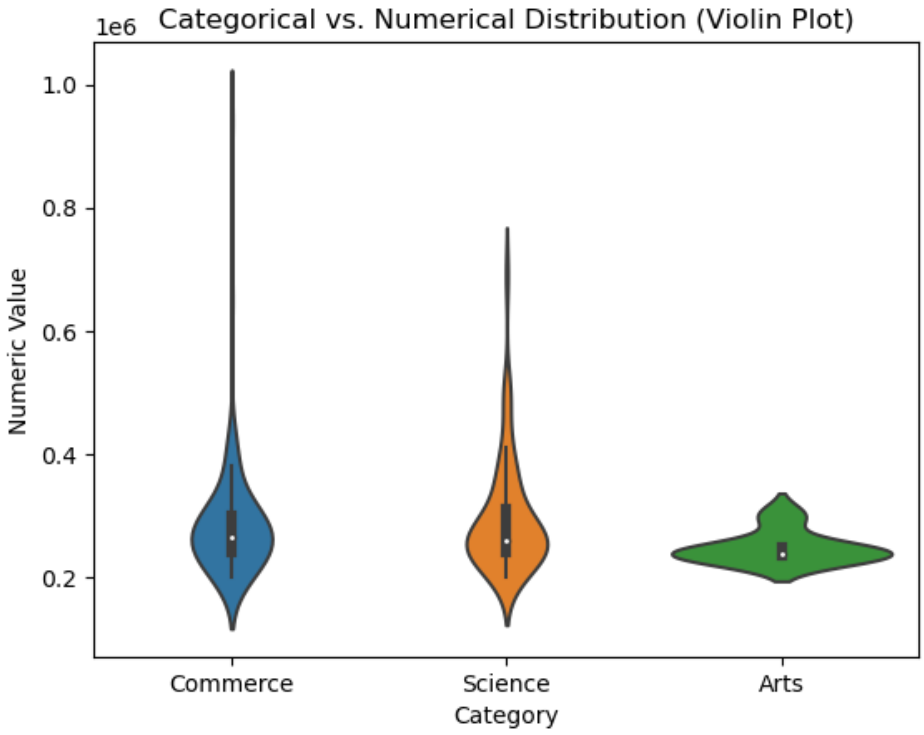
In [31]:
```
sns.barplot(x='hsc_s', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



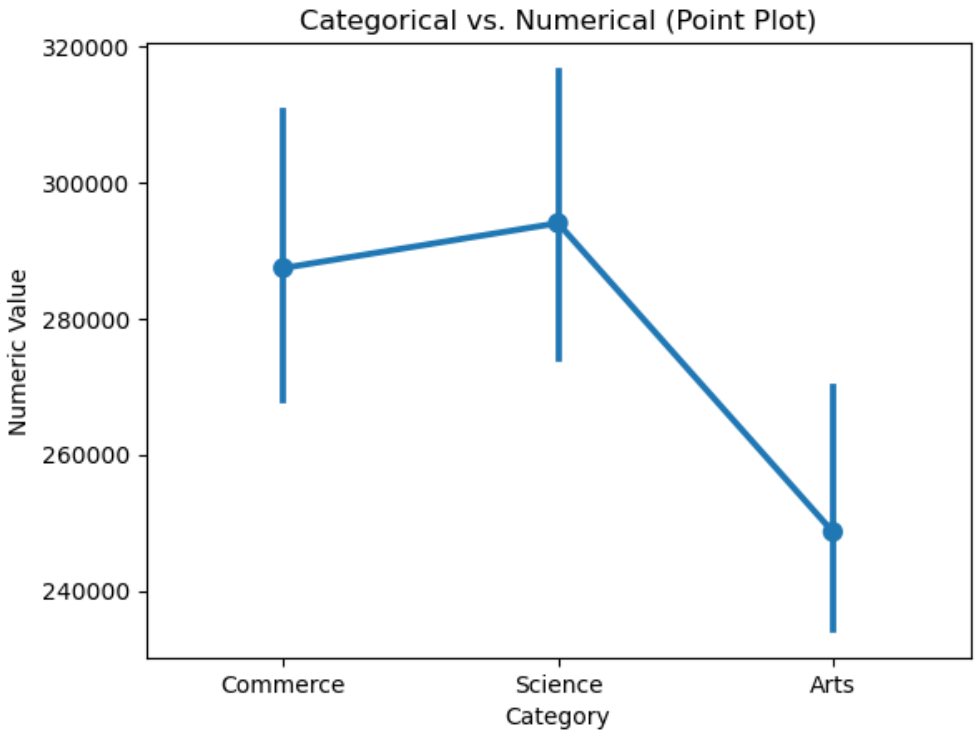
In [32]:
```
sns.boxplot(x='hsc_s', y='salary', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [33]:
```
sns.violinplot(x='hsc_s', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```



In [34]:
```
sns.pointplot(x='hsc_s', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

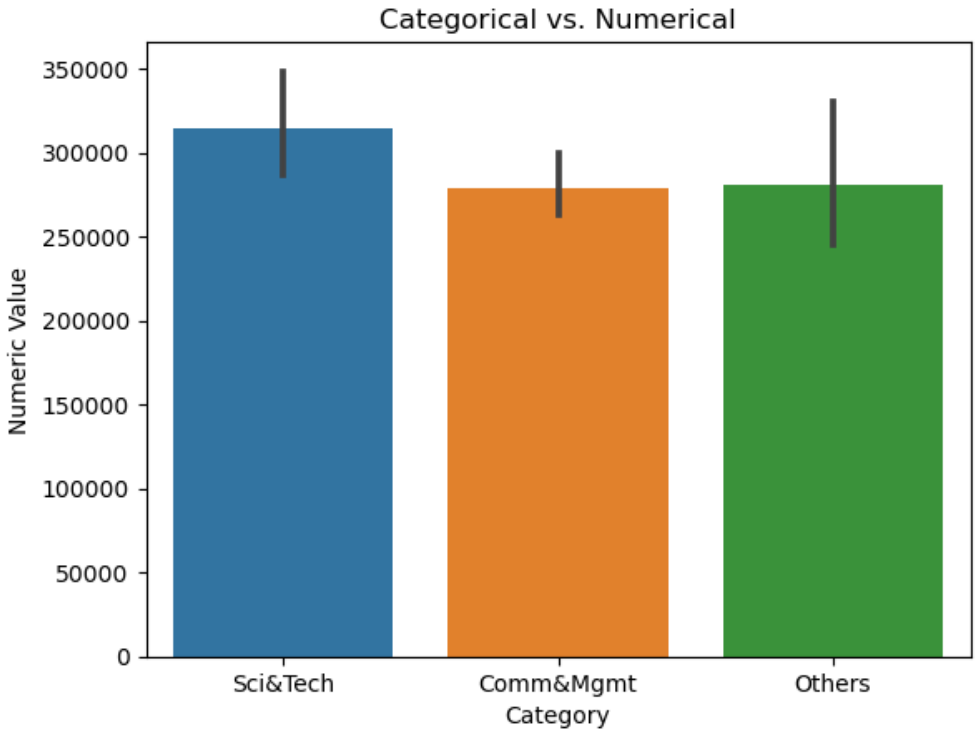


In [ ]:
```
```

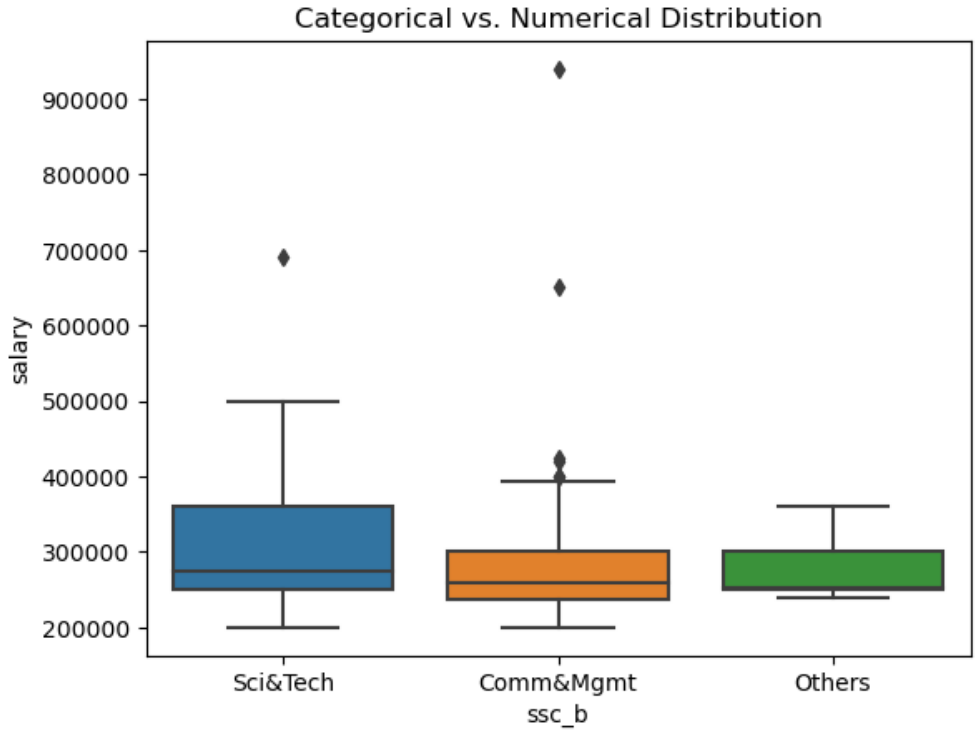
degree_t

In [35]:
```
sns.barplot(x='degree_t', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```

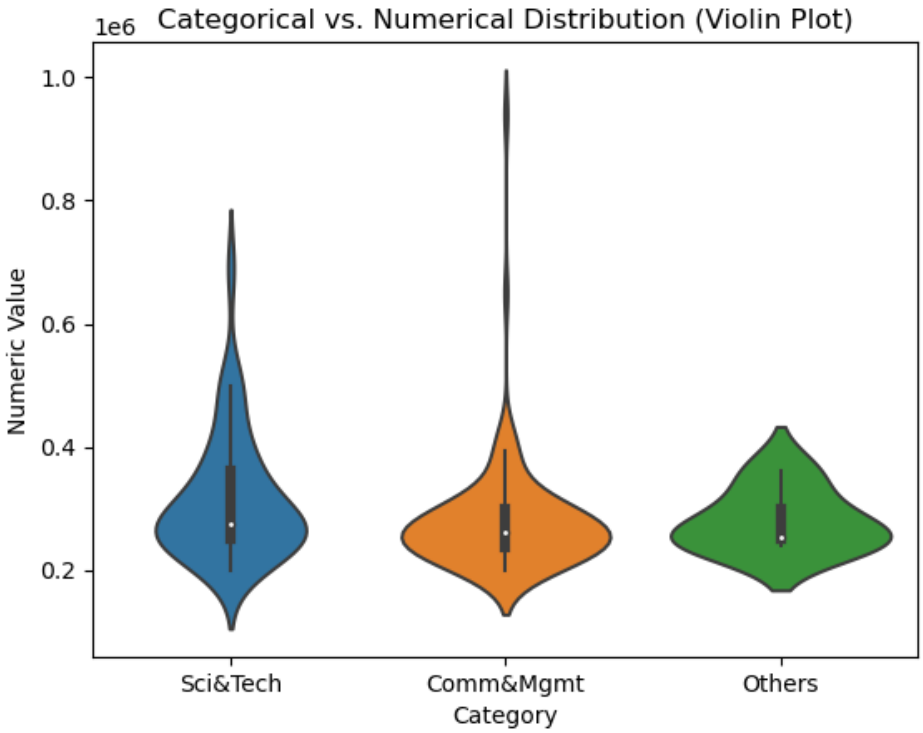
```
C:\Users\kukku\anaconda3\lib\site-packages\seaborn\algorithms.py:98: RuntimeWarning: Mean of empty slic
e
  boot_dist.append(f(*sample, **func_kwargs))
```



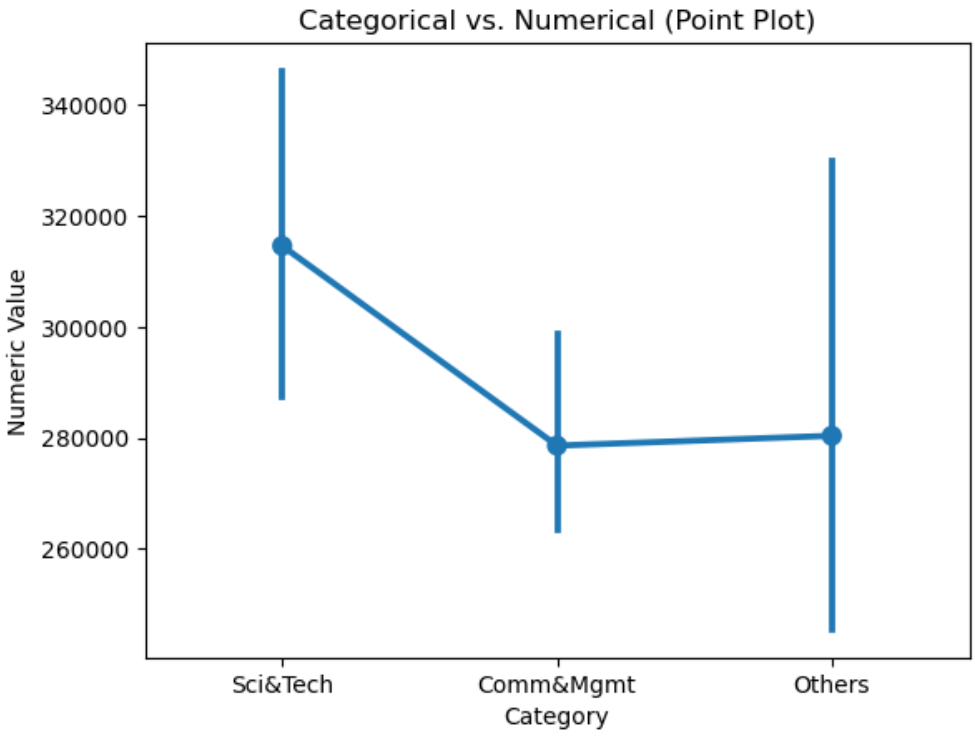
In [36]:
```
sns.boxplot(x='degree_t', y='salary', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [37]:
```
sns.violinplot(x='degree_t', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

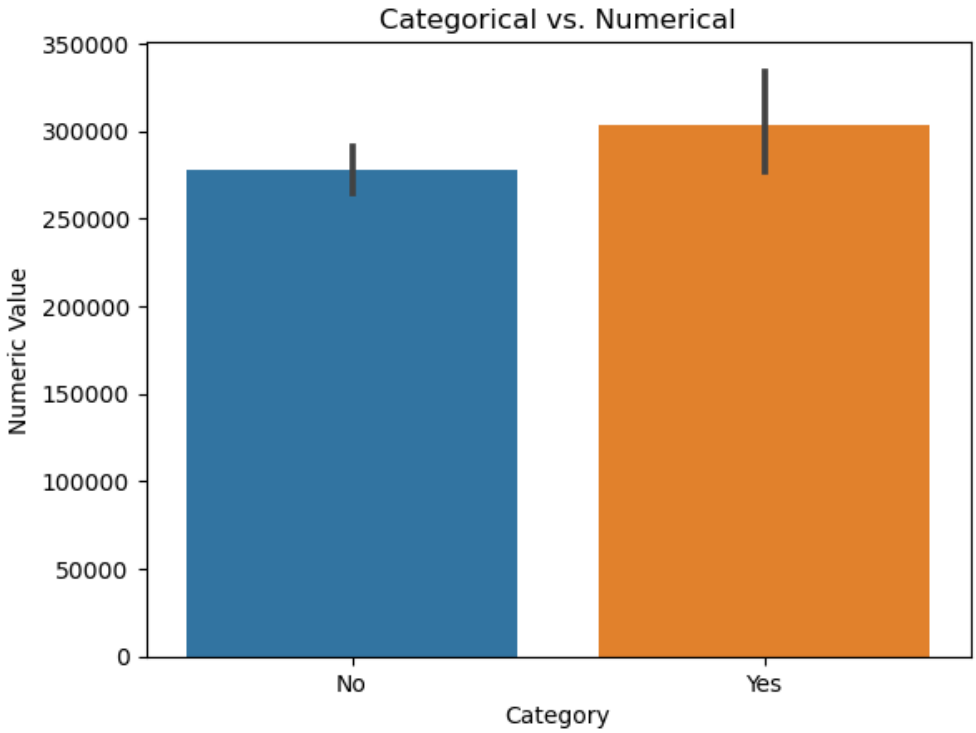


In [38]:
```
sns.pointplot(x='degree_t', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

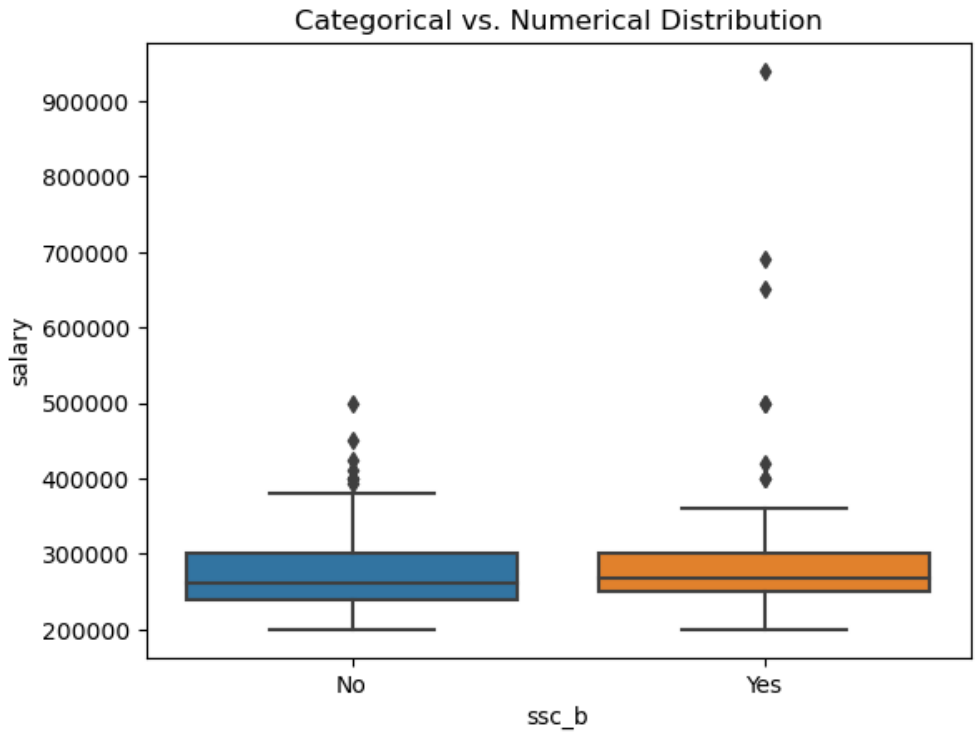


workex

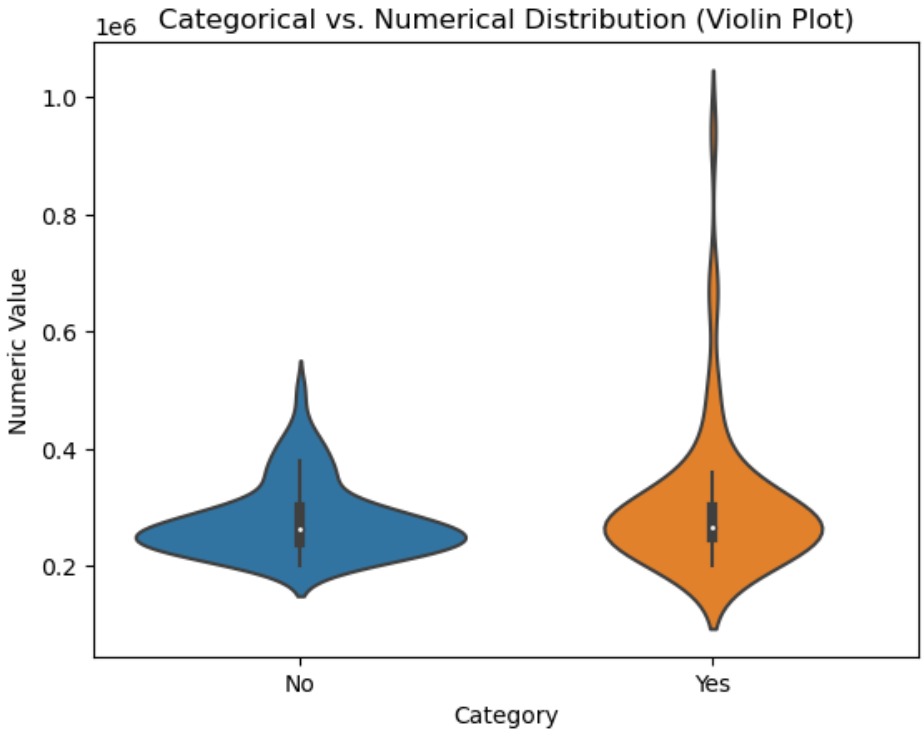
In [39]:
```
sns.barplot(x='workex', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



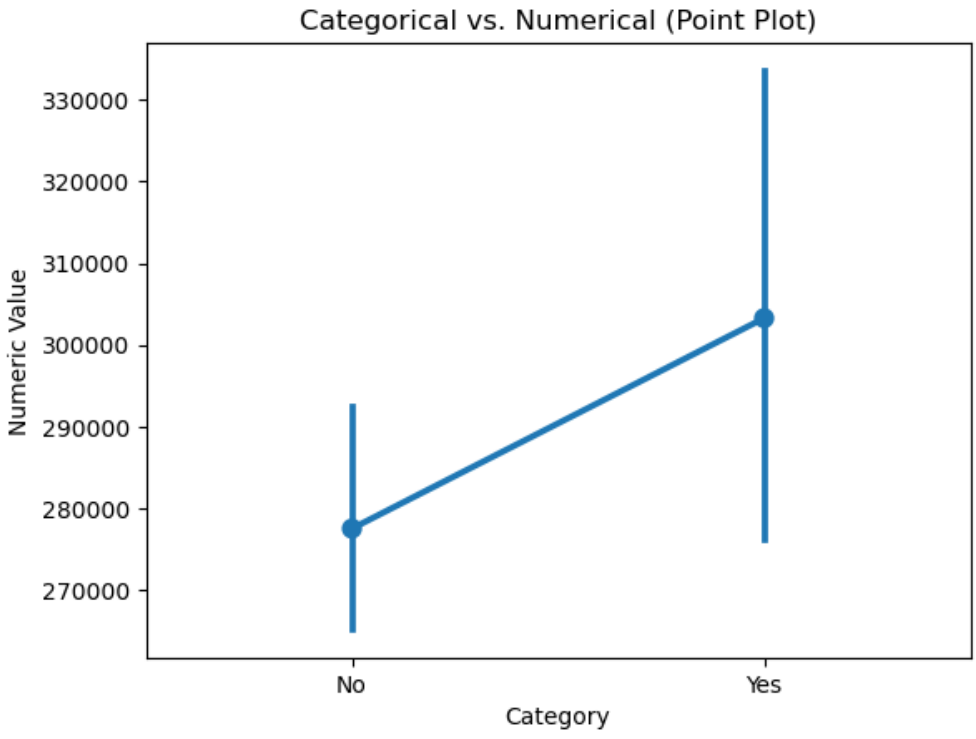
In [40]:
```
sns.boxplot(x='workex', y='salary', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [41]:
```
sns.violinplot(x='workex', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

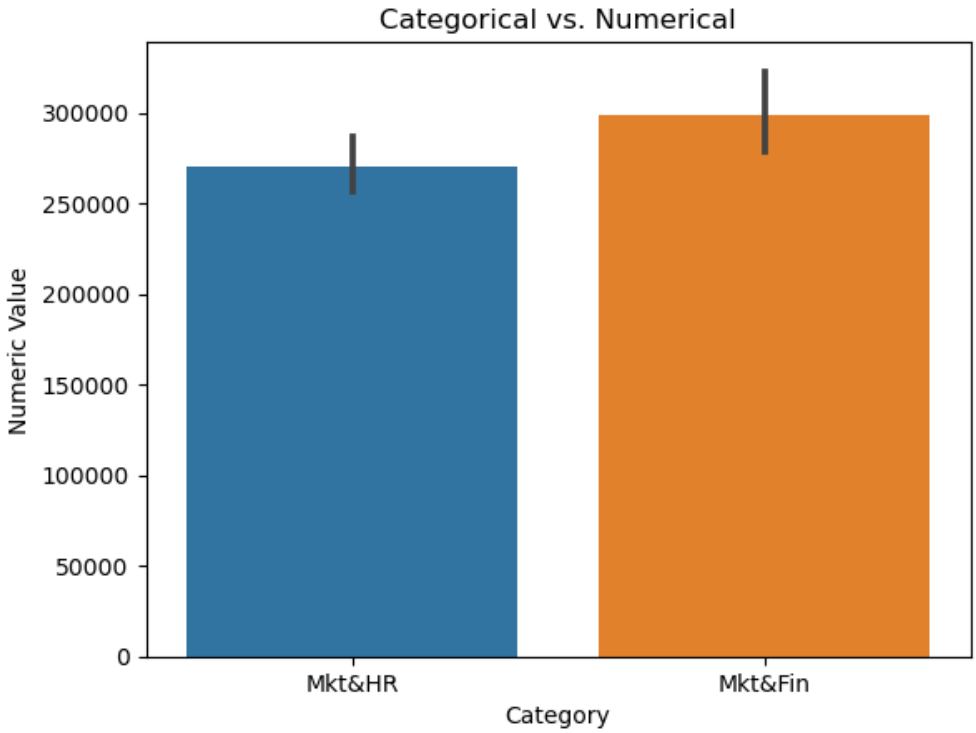


In [42]:
```
sns.pointplot(x='workex', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

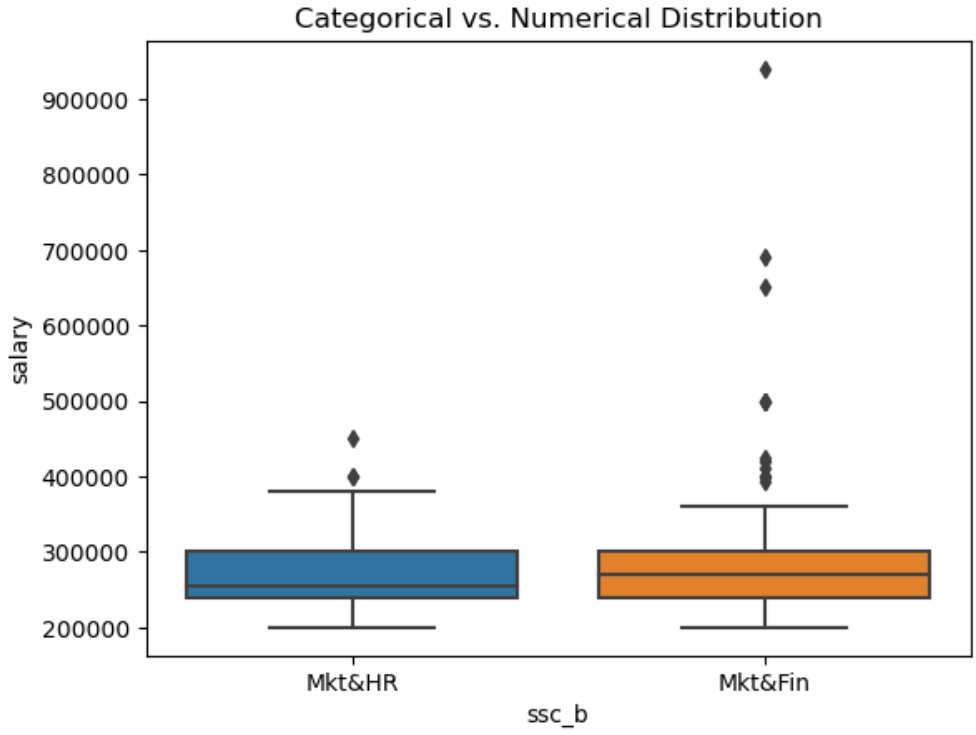


specialisation

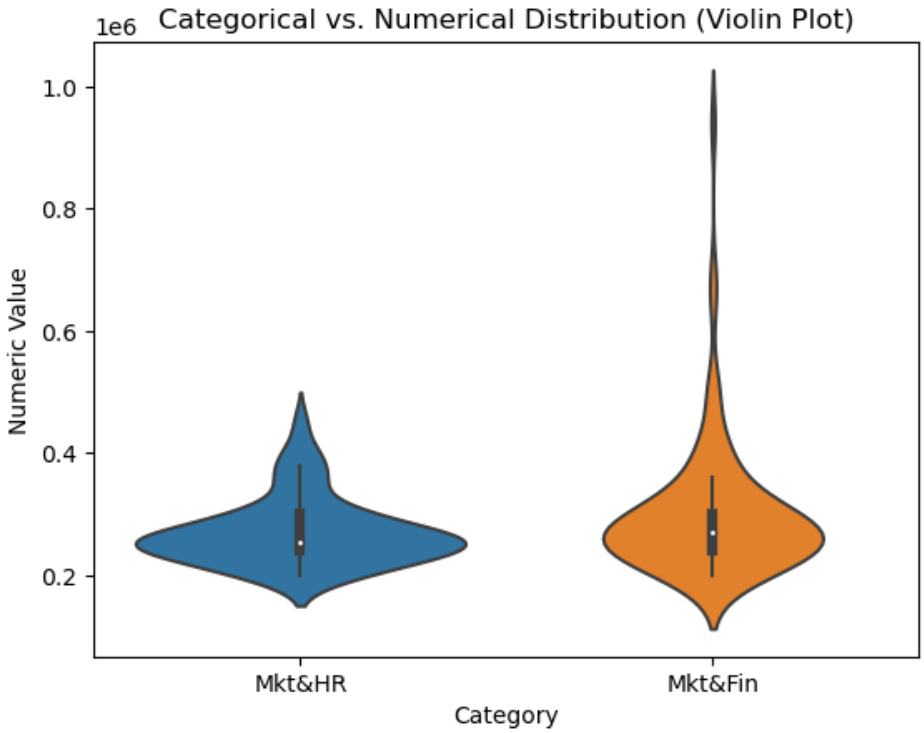
In [43]:
```
sns.barplot(x='specialisation', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



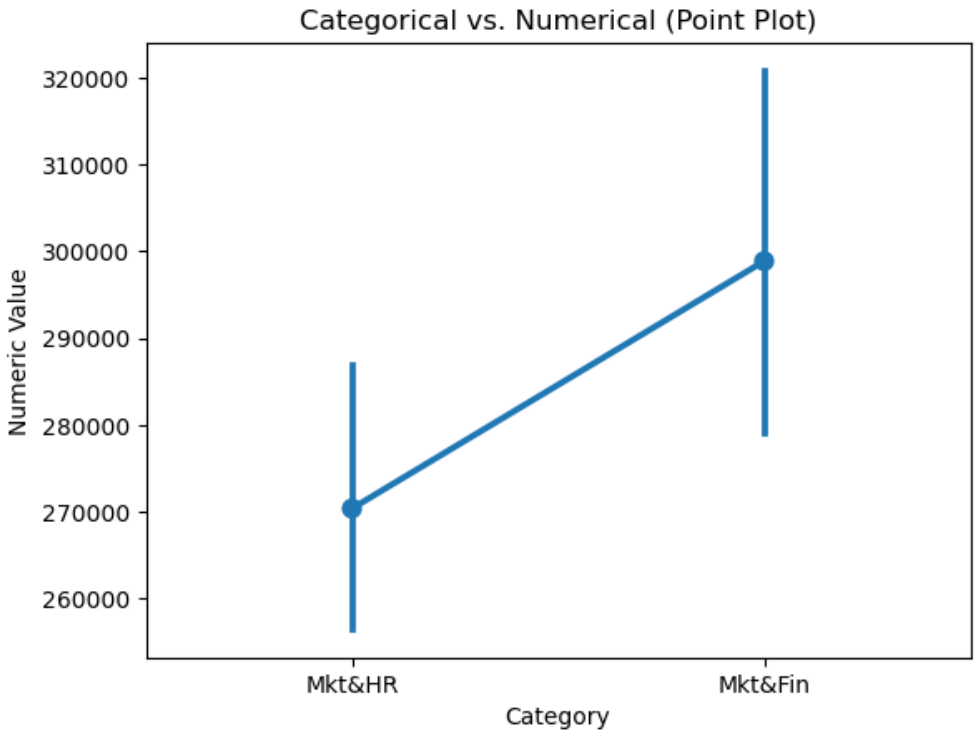
In [44]:
```
sns.boxplot(x='specialisation', y='salary', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [45]:
```
sns.violinplot(x='specialisation', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```



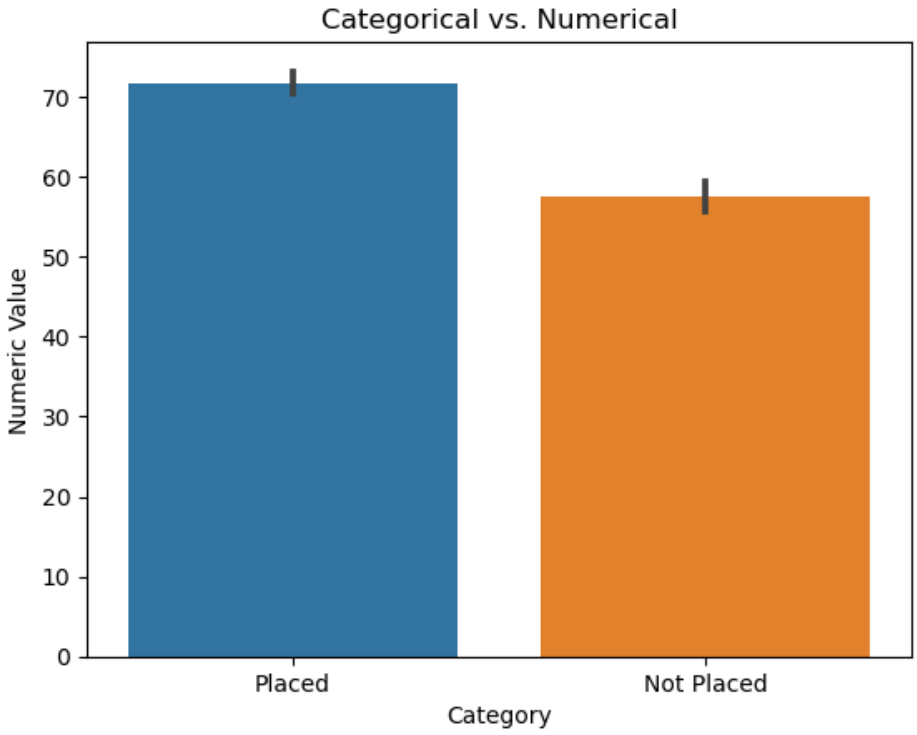
In [46]:
```
sns.pointplot(x='specialisation', y='salary', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```



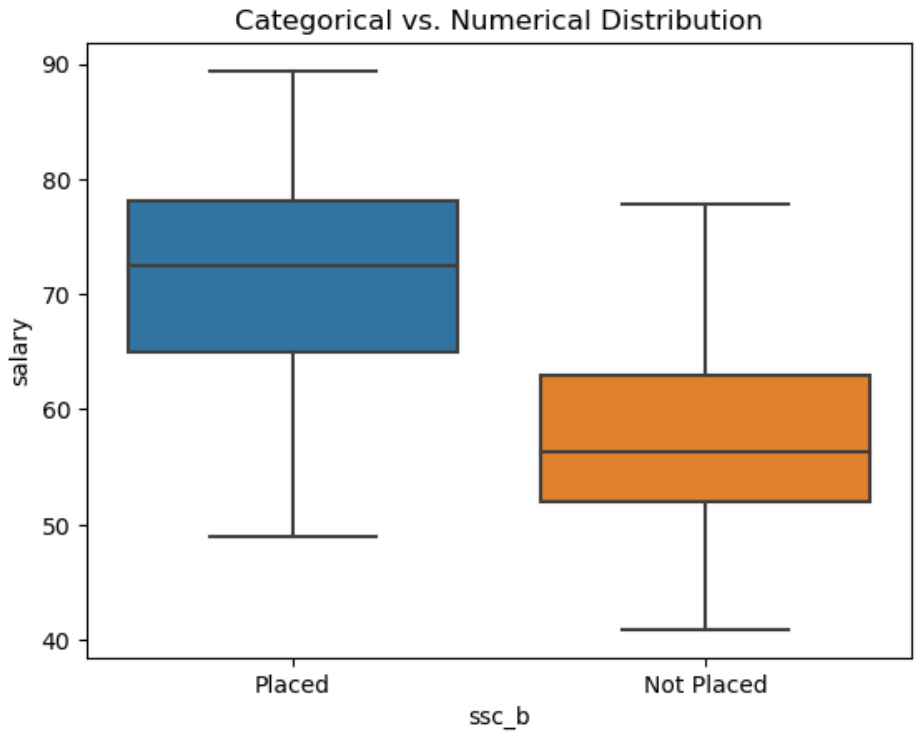
NUmericcal with placed

ssc_p

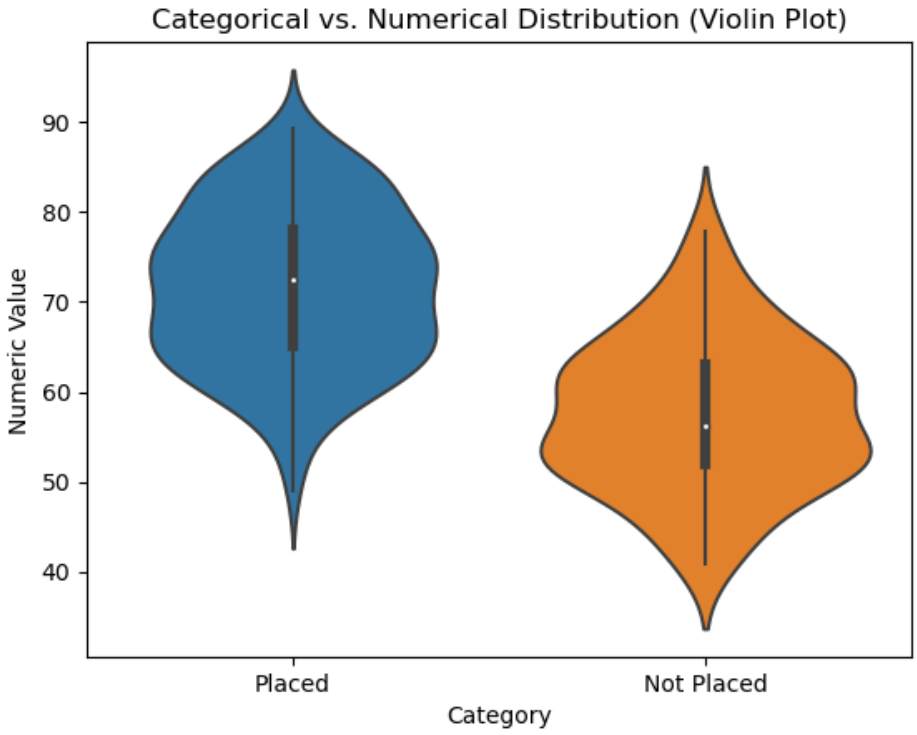
In [47]:
```
sns.barplot(x='status', y='ssc_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



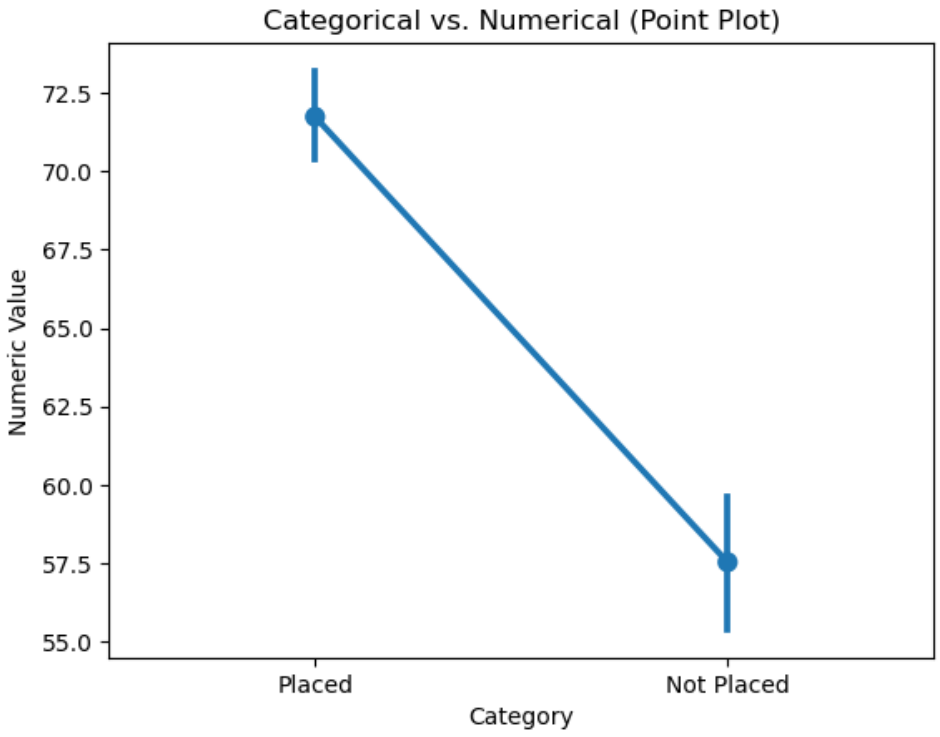
In [48]:
```
sns.boxplot(x='status', y='ssc_p', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [49]:
```
sns.violinplot(x='status', y='ssc_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

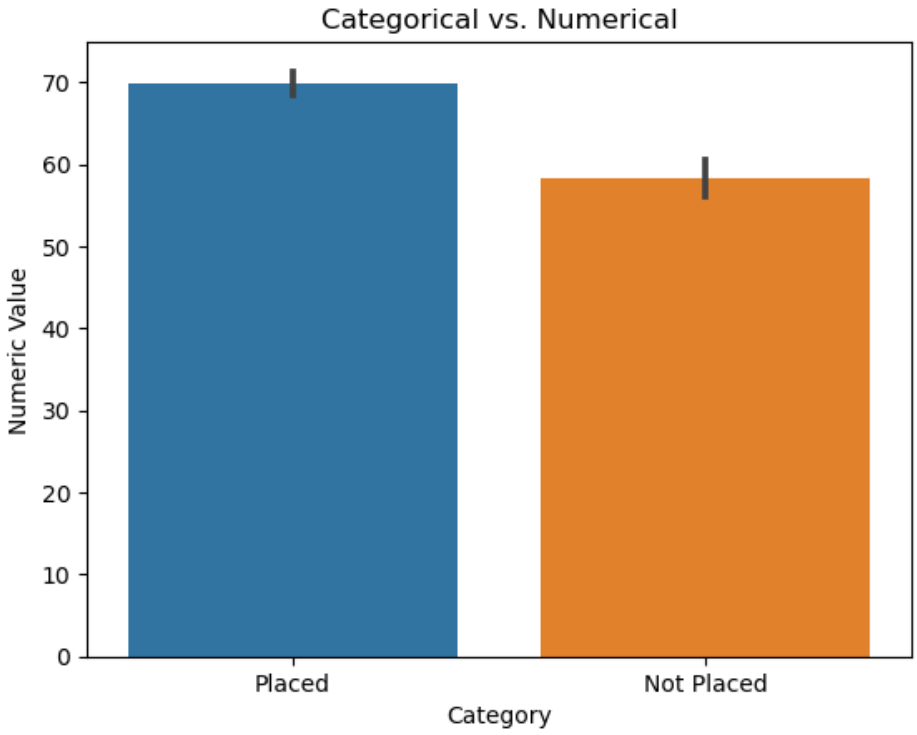


In [50]:
```
sns.pointplot(x='status', y='ssc_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

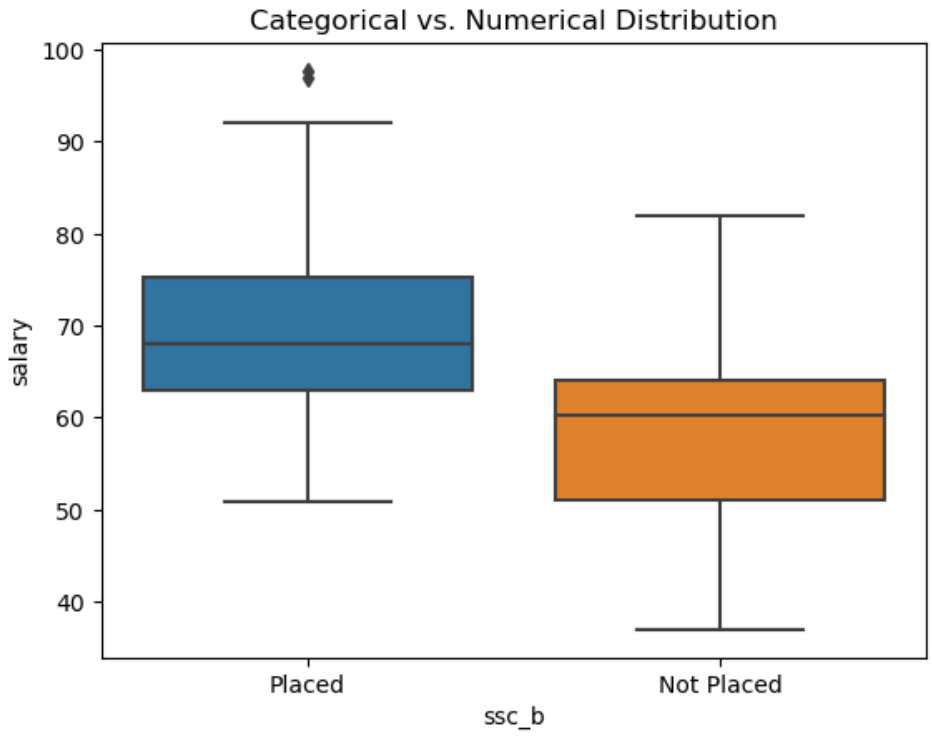


hsc_p

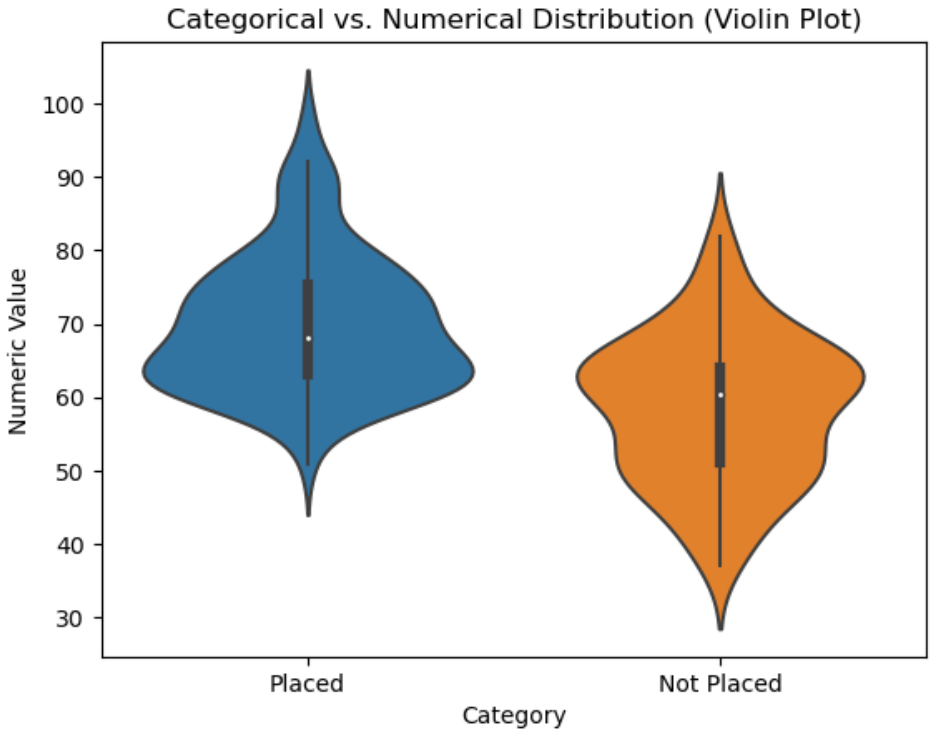
In [51]:
```
sns.barplot(x='status', y='hsc_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



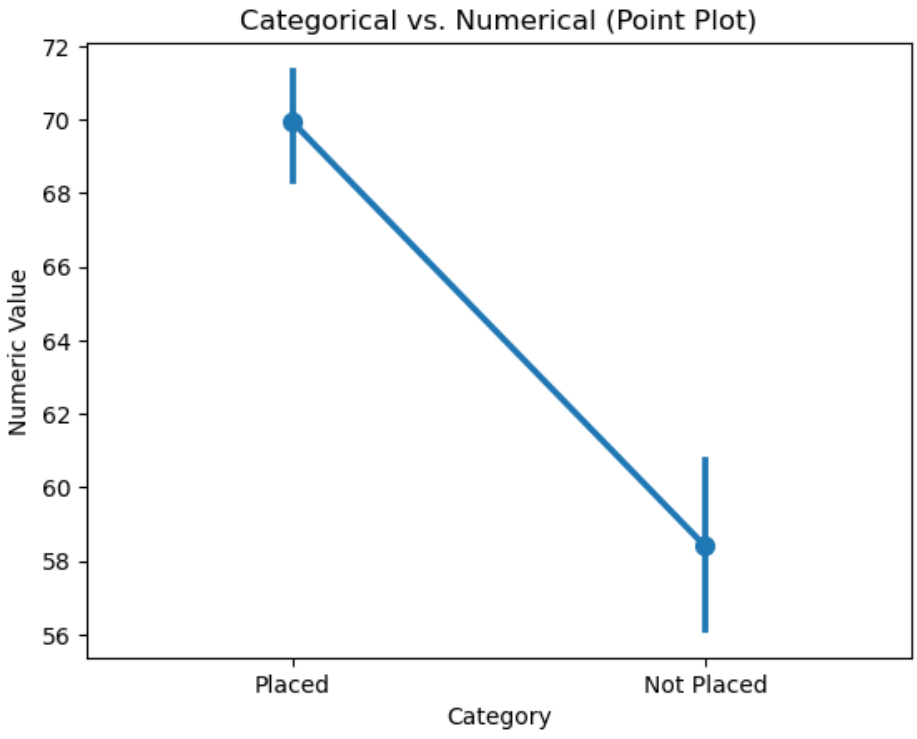
In [52]:
```
sns.boxplot(x='status', y='hsc_p', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [53]:
```
sns.violinplot(x='status', y='hsc_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

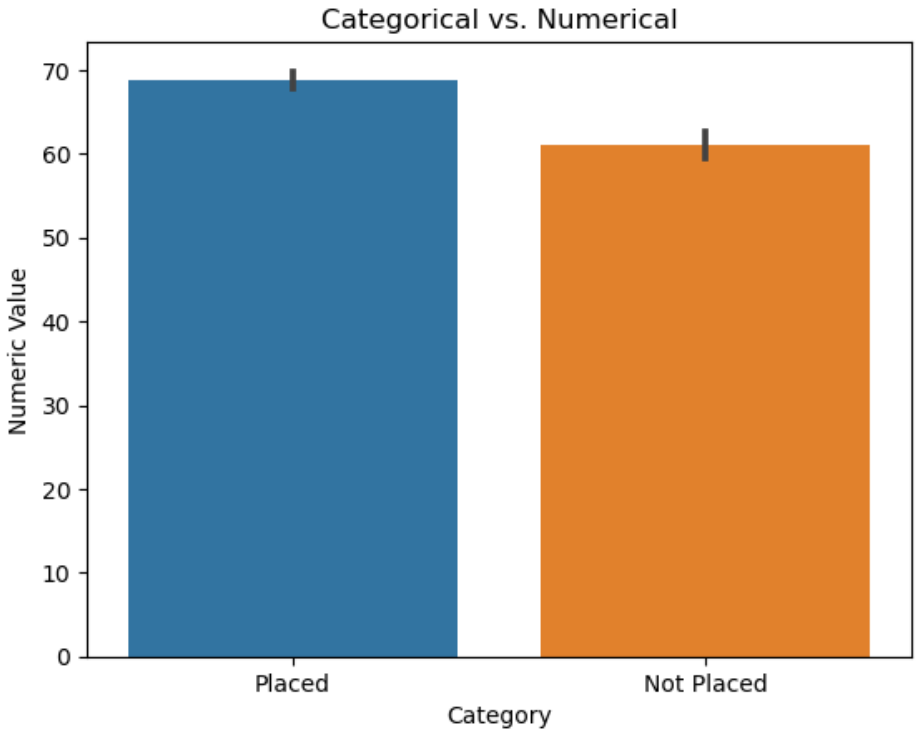


In [54]:
```
sns.pointplot(x='status', y='hsc_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

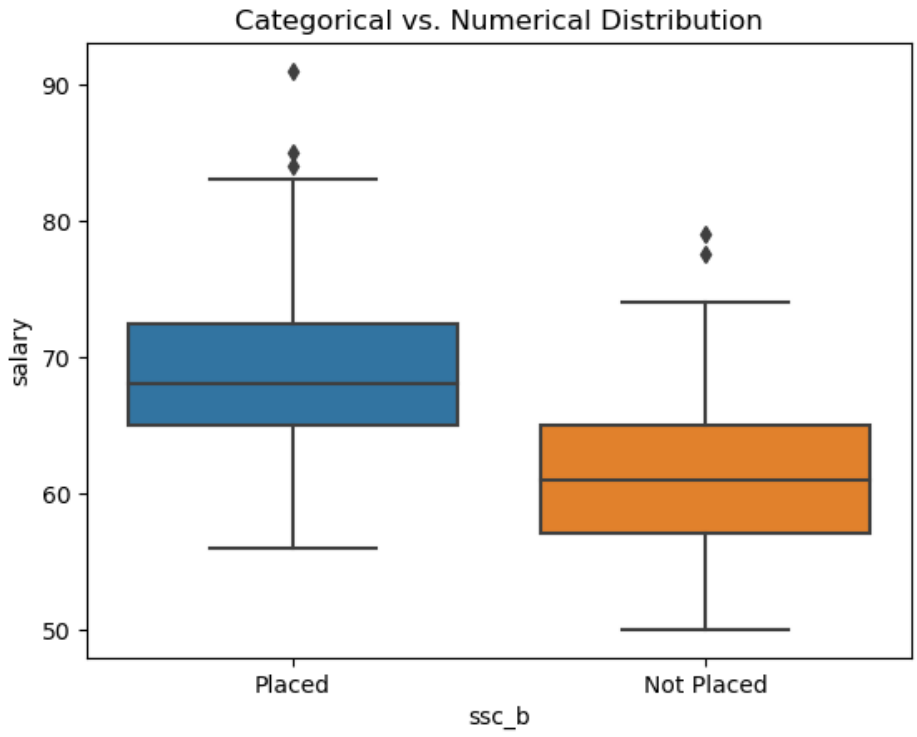


degree_p

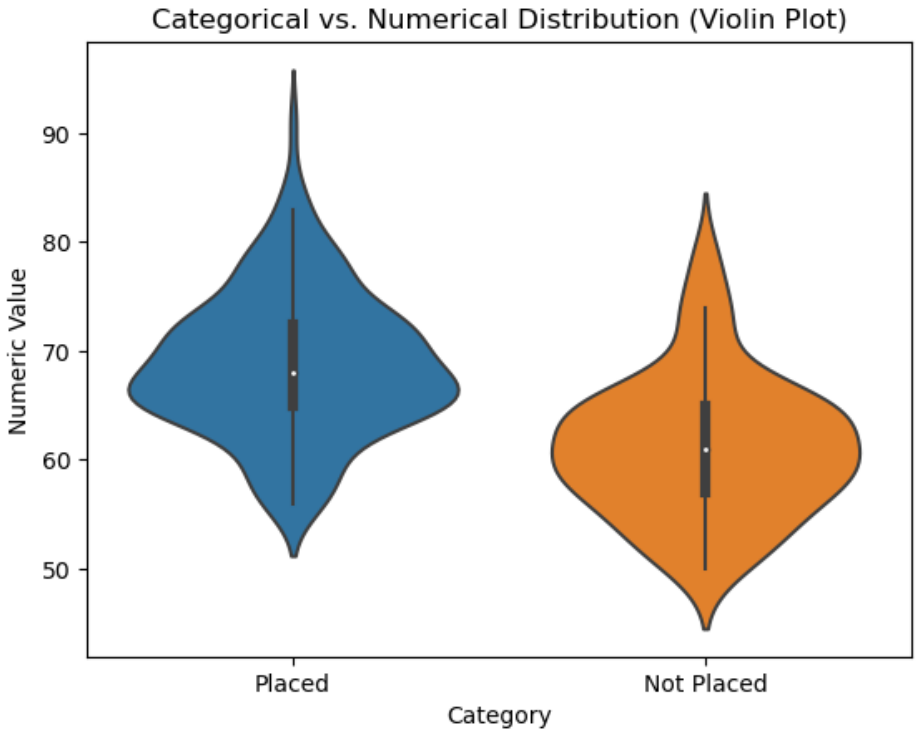
In [55]:
```
sns.barplot(x='status', y='degree_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



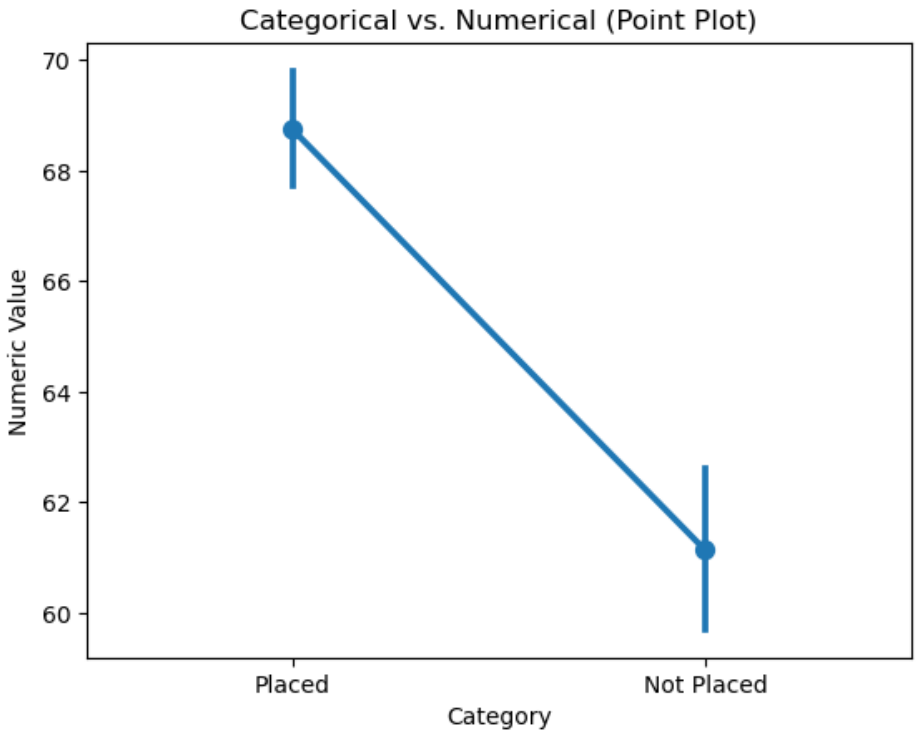
In [56]:
```
sns.boxplot(x='status', y='degree_p', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [57]:
```
sns.violinplot(x='status', y='degree_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

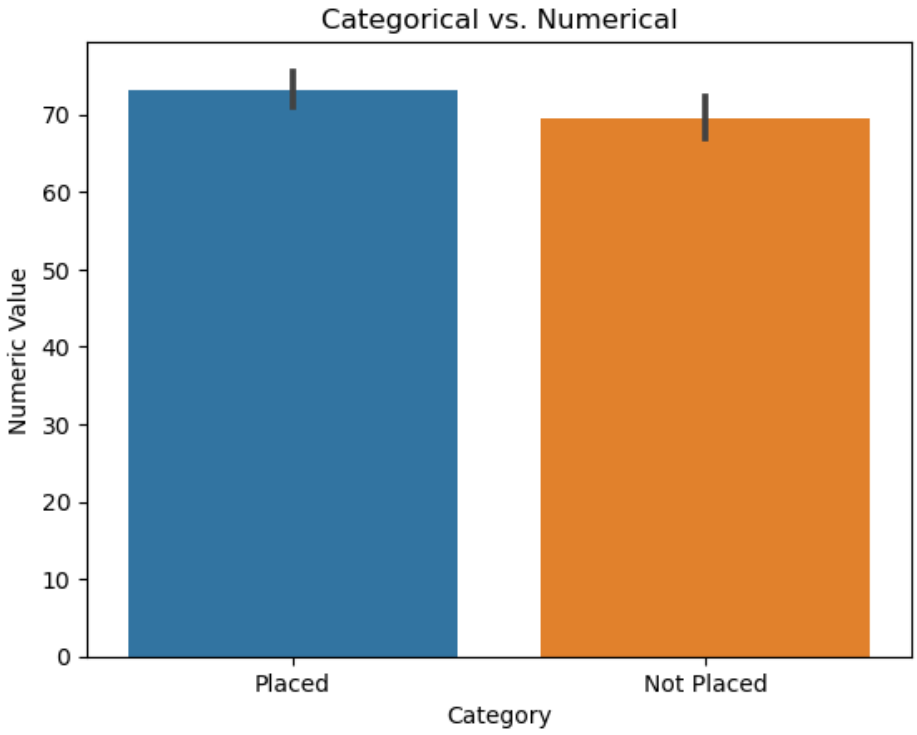


In [58]:
```
sns.pointplot(x='status', y='degree_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

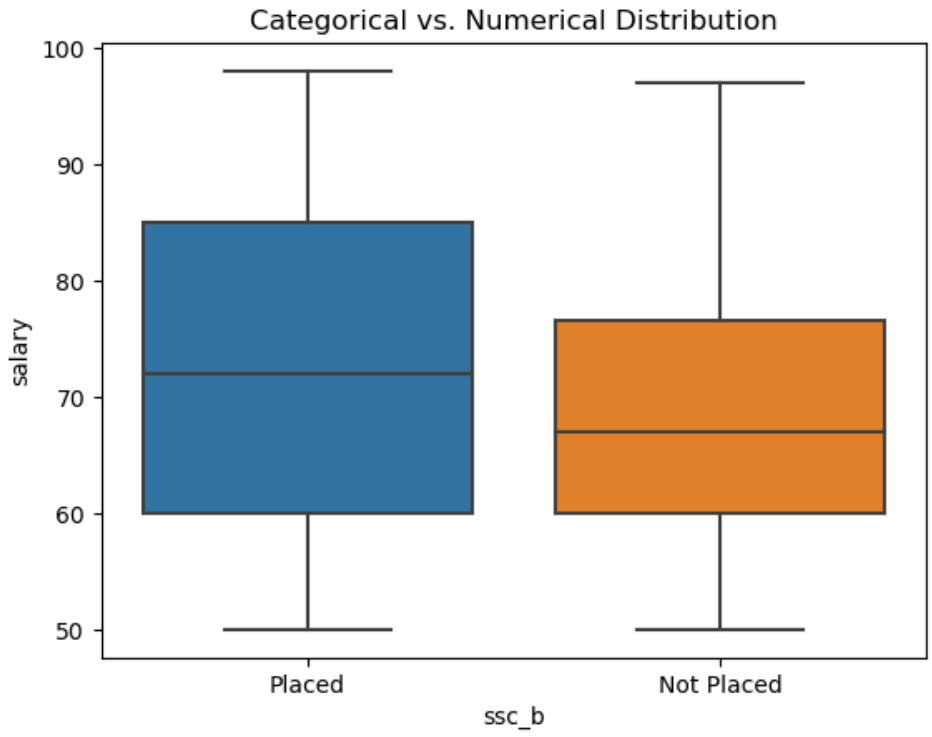


etest_p

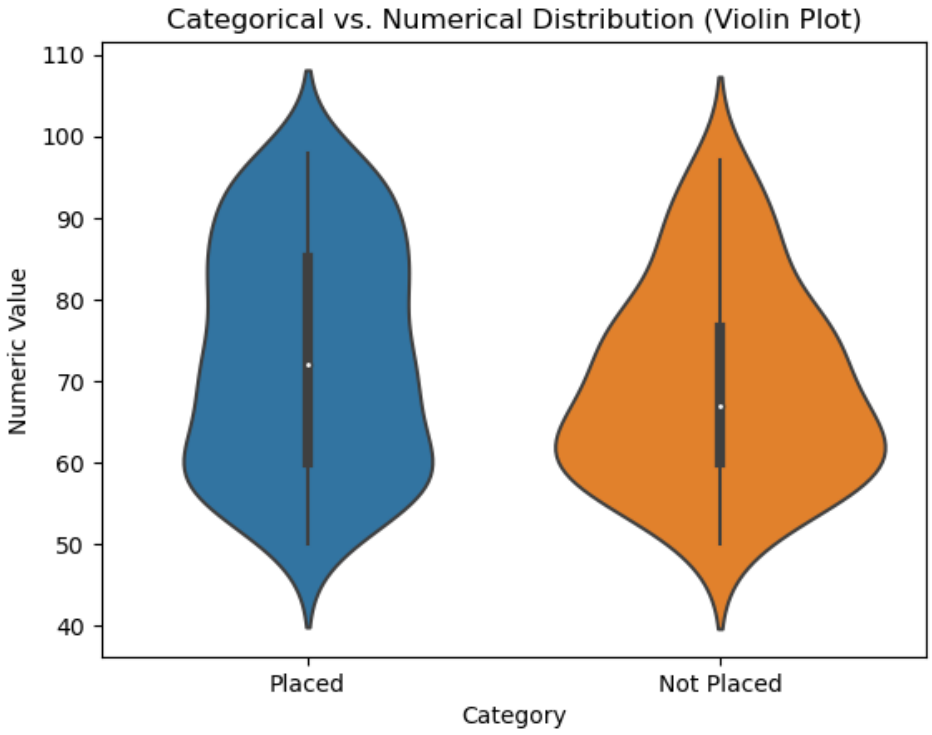
In [59]:
```
sns.barplot(x='status', y='etest_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



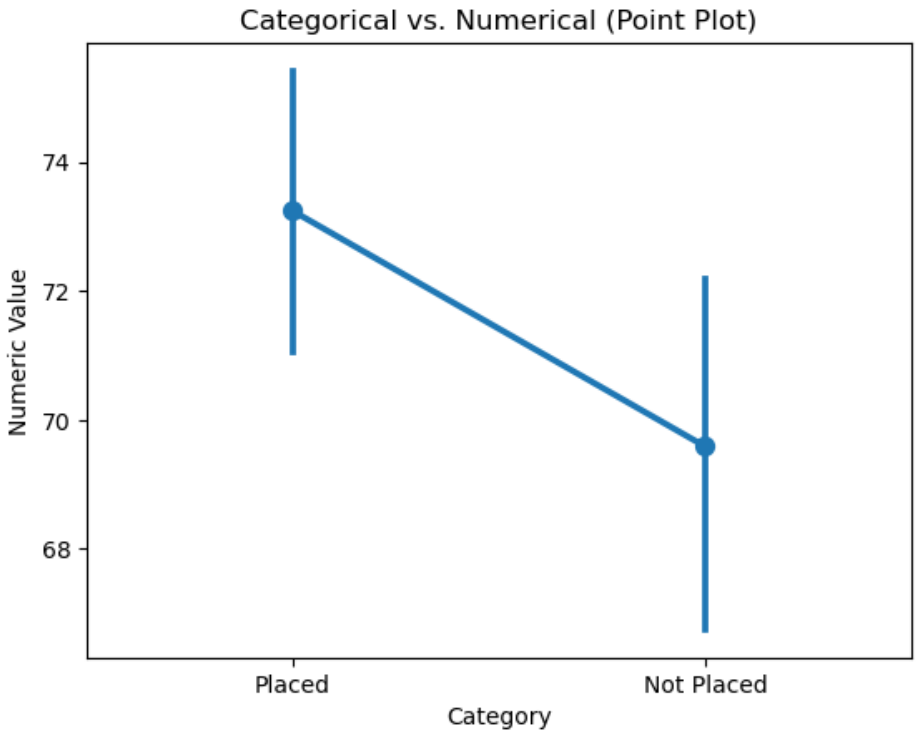
In [60]:
```
sns.boxplot(x='status', y='etest_p', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [61]:
```
sns.violinplot(x='status', y='etest_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```

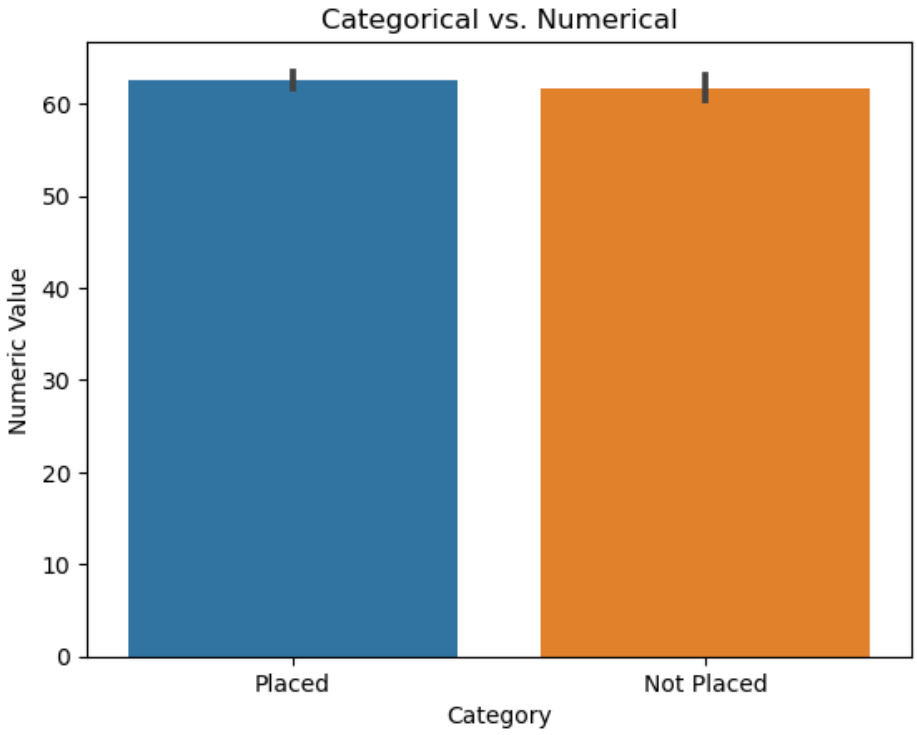


In [62]:
```
sns.pointplot(x='status', y='etest_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

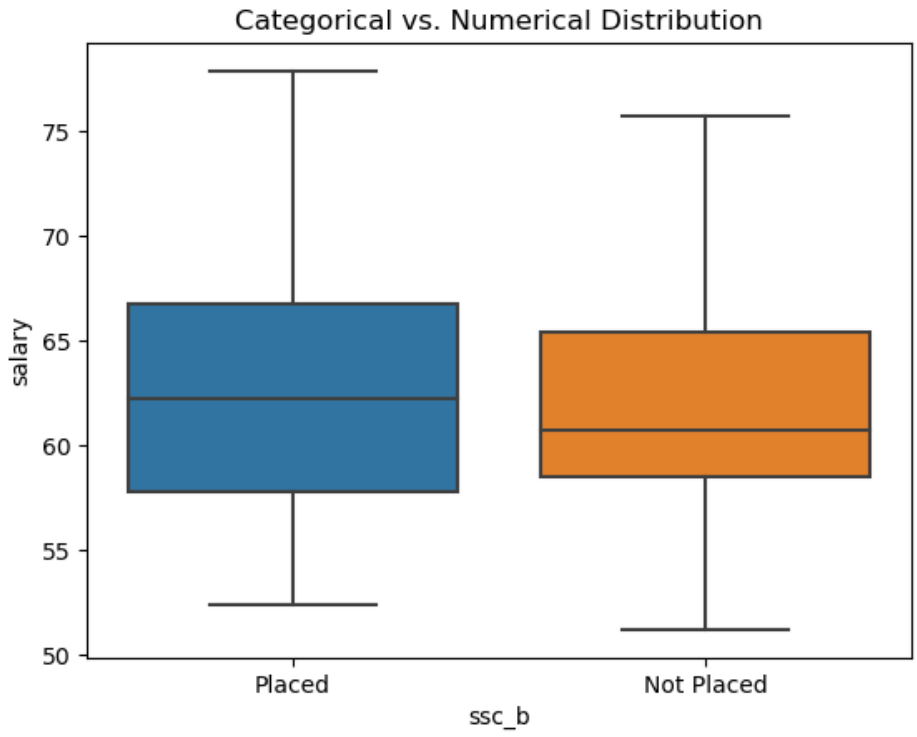


mba_p

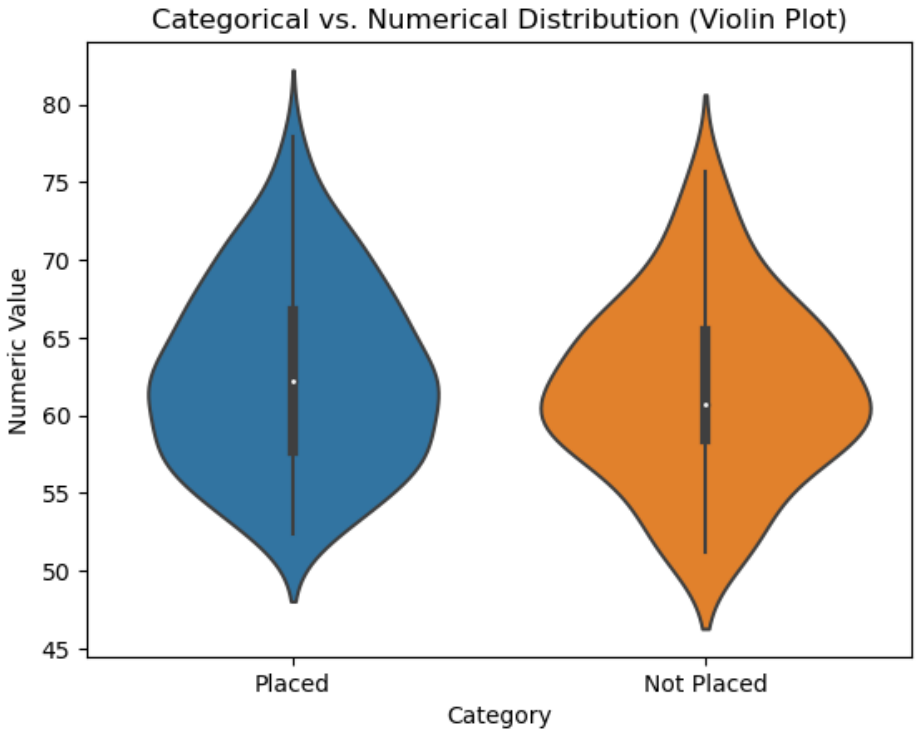
In [63]:
```
sns.barplot(x='status', y='mba_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical')
plt.show()
```



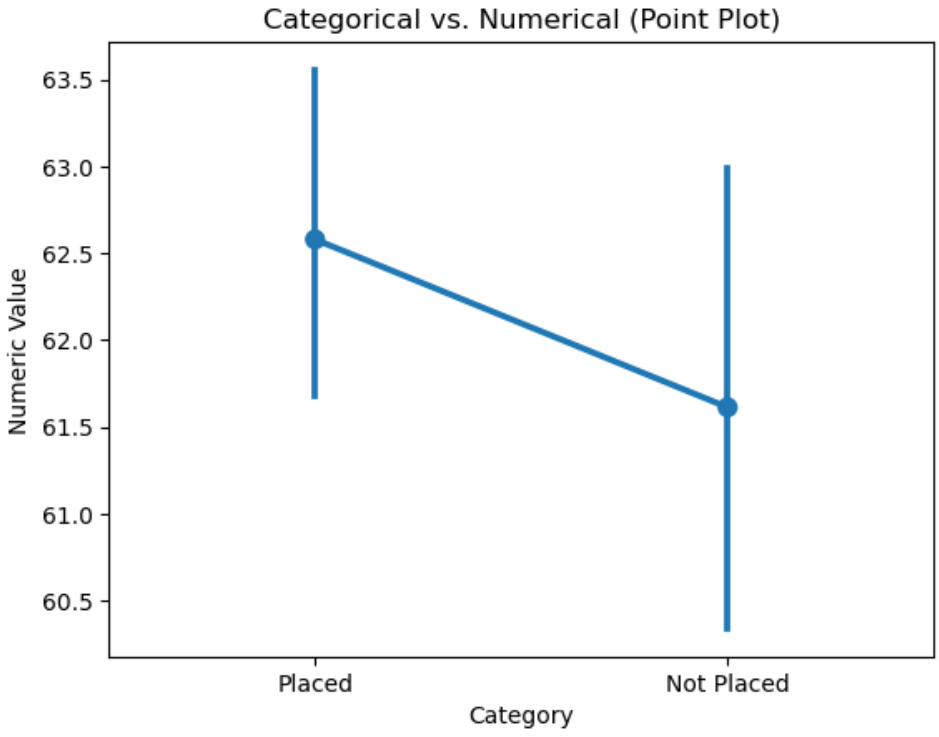
In [64]:
```
sns.boxplot(x='status', y='mba_p', data=df)
plt.xlabel('ssc_b')
plt.ylabel('salary')
plt.title('Categorical vs. Numerical Distribution')
plt.show()
```



In [65]:
```
sns.violinplot(x='status', y='mba_p', data=df)
plt.xlabel('Category')
plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical Distribution (Violin Plot)')
plt.show()
```



In [66]:
```
sns.pointplot(x='status', y='mba_p', data=df)

plt.xlabel('Category')

plt.ylabel('Numeric Value')
plt.title('Categorical vs. Numerical (Point Plot)')
plt.show()
```

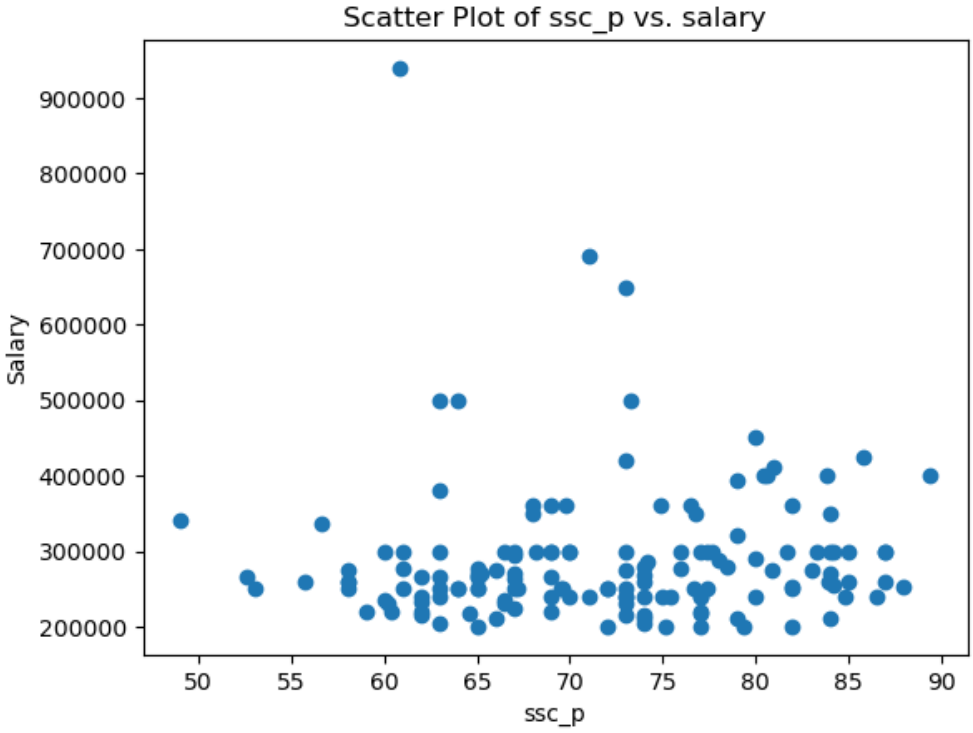


In [ ]:
```
```

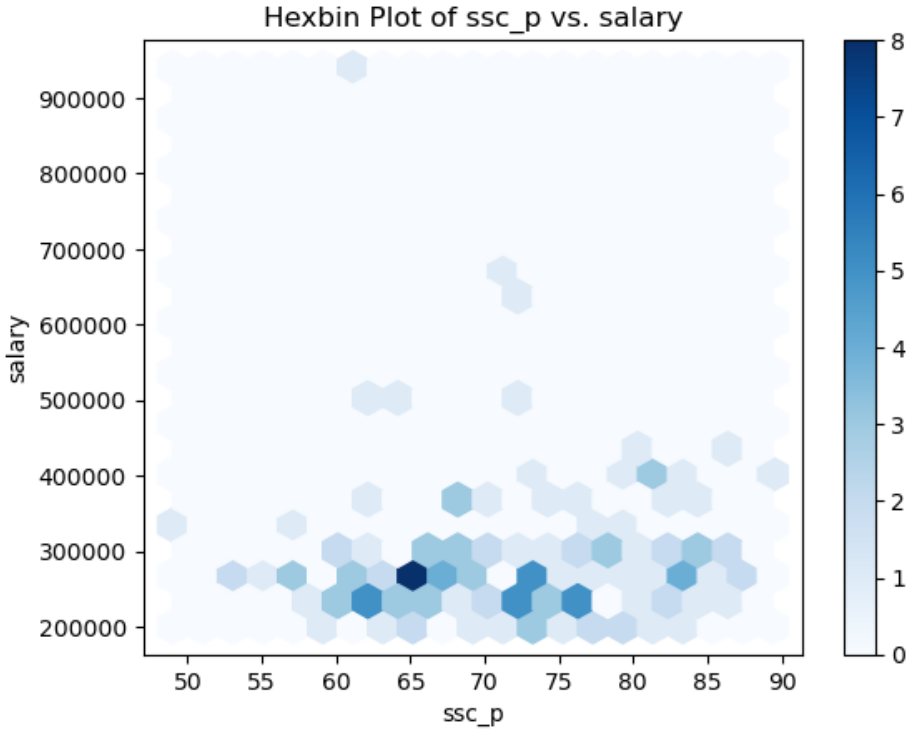
Numerical vs salary

ssc_p

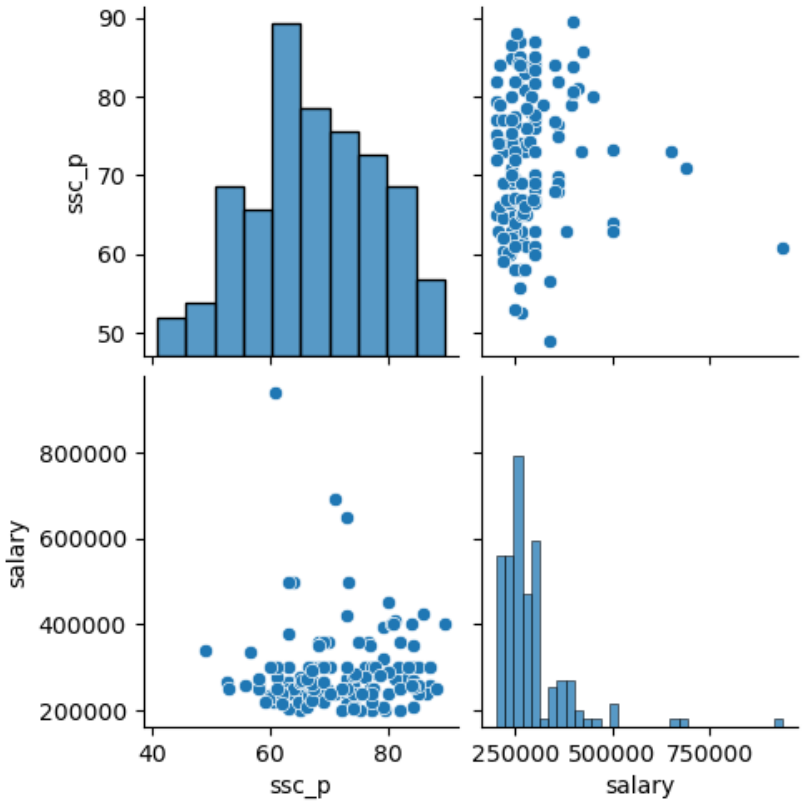
In [67]:
```
plt.scatter(df['ssc_p'], df['salary'])
plt.xlabel('ssc_p')
plt.ylabel('Salary')
plt.title('Scatter Plot of ssc_p vs. salary')
plt.show()
```



In [68]:
```
plt.hexbin(df['ssc_p'], df['salary'], gridsize=20, cmap='Blues')
plt.xlabel('ssc_p')
plt.ylabel('salary')
plt.title('Hexbin Plot of ssc_p vs. salary')
plt.colorbar()
plt.show()
```



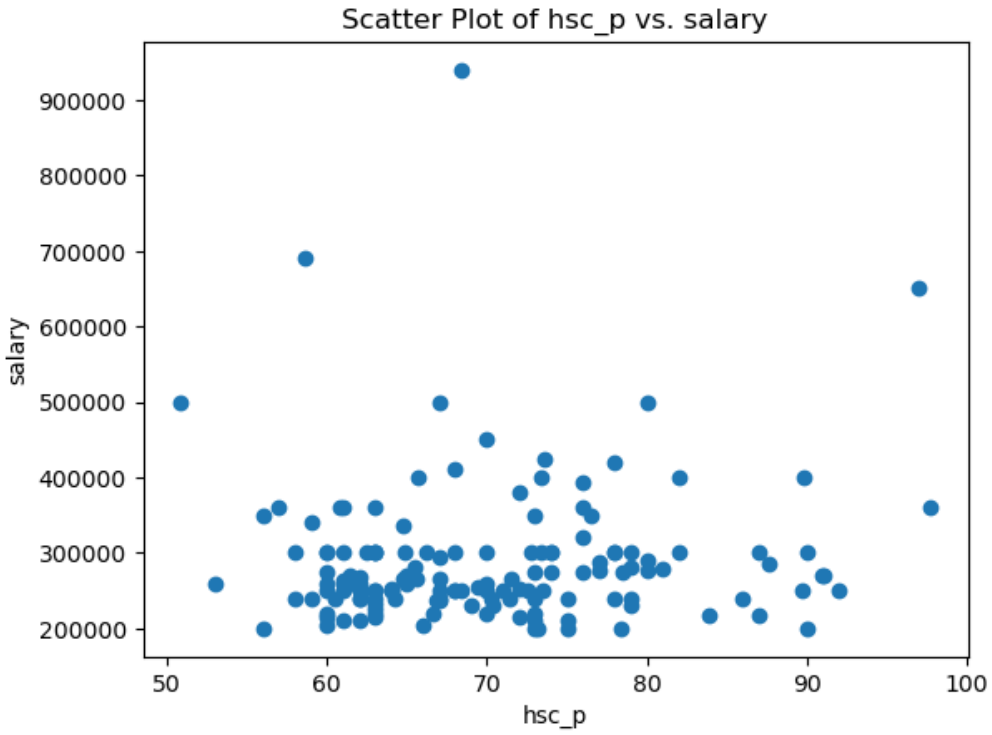
In [69]:
```
sns.pairplot(df[['ssc_p', 'salary']])
plt.show()
```



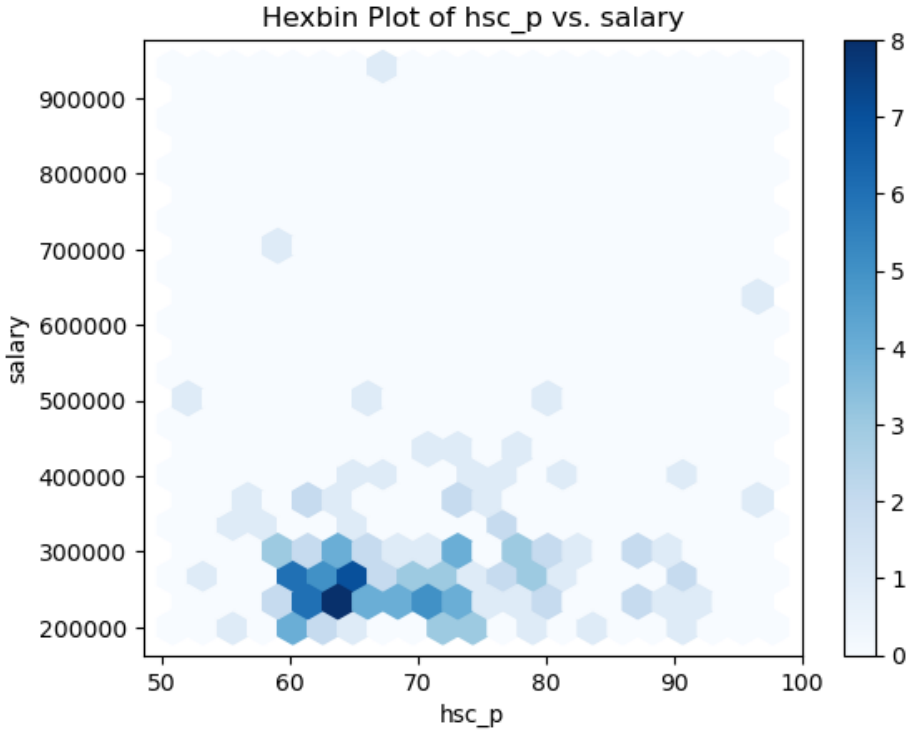
In [70]:
```
plt.plot(df['ssc_p'], df['salary'])
plt.xlabel('ssc_p')
plt.ylabel('salary')
plt.title('Line Plot of ssc_p vs. salary')
plt.show()
```

hsc_p

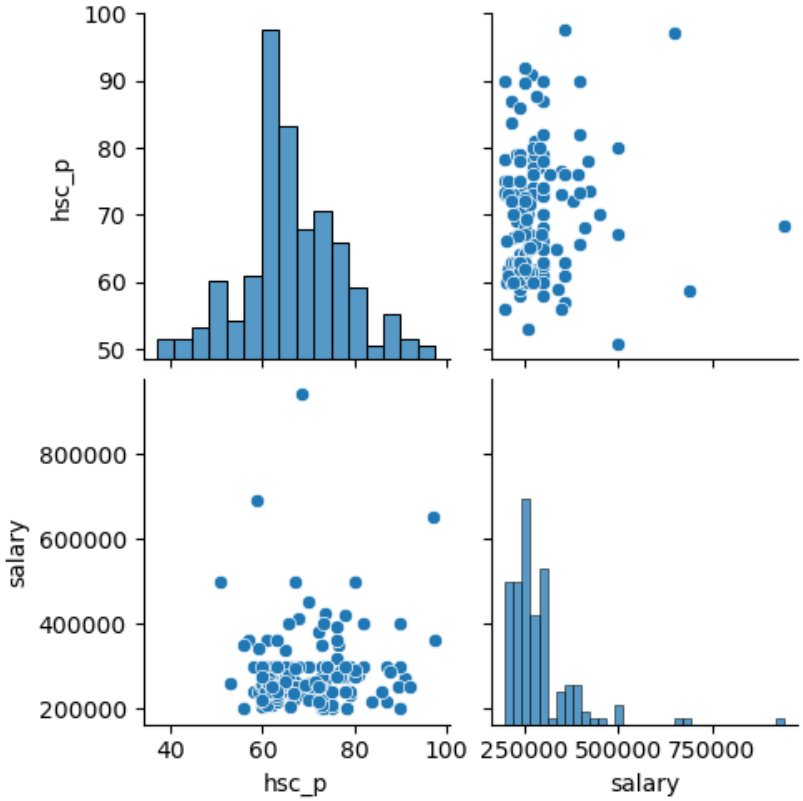
In [71]:
```
plt.scatter(df['hsc_p'], df['salary'])
plt.xlabel('hsc_p')
plt.ylabel('salary')
plt.title('Scatter Plot of hsc_p vs. salary')
plt.show()
```



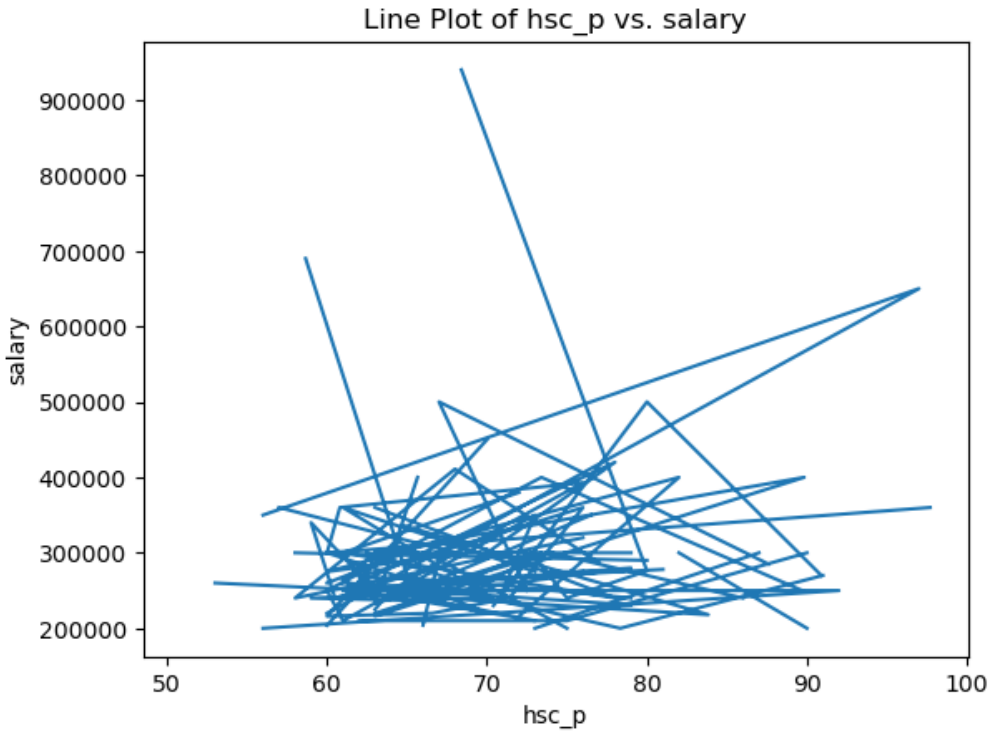
In [72]:
```
plt.hexbin(df['hsc_p'], df['salary'], gridsize=20, cmap='Blues')
plt.xlabel('hsc_p')
plt.ylabel('salary')
plt.title('Hexbin Plot of hsc_p vs. salary')
plt.colorbar()
plt.show()
```



In [73]:
```
sns.pairplot(df[['hsc_p', 'salary']])
plt.show()
```



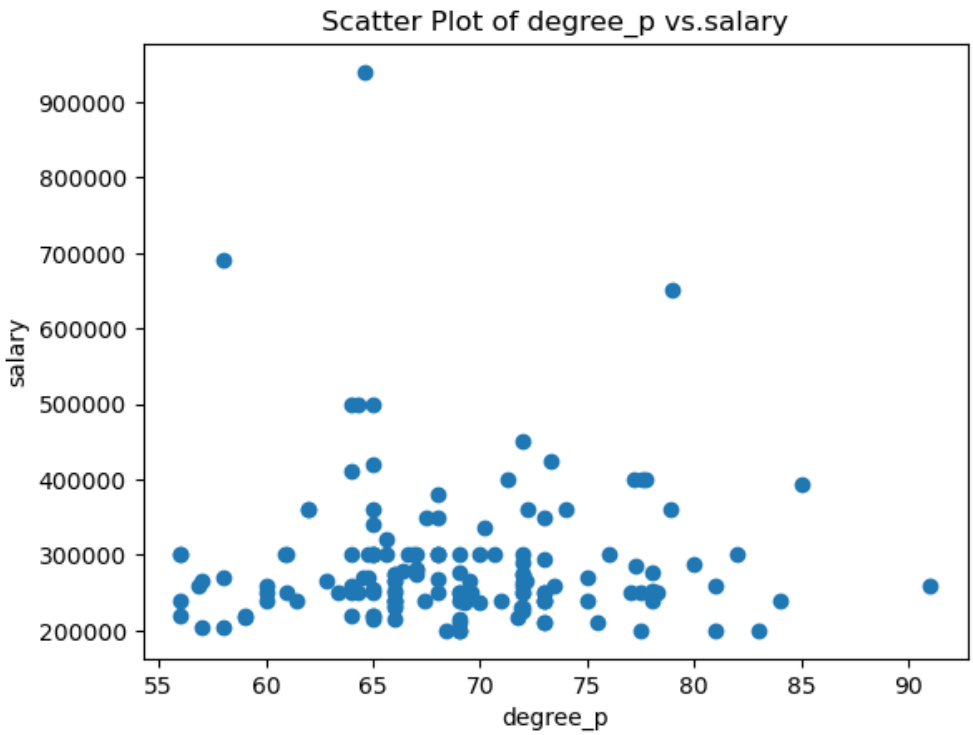
In [74]:
```
plt.plot(df['hsc_p'], df['salary'])
plt.xlabel('hsc_p')
plt.ylabel('salary')
plt.title('Line Plot of hsc_p vs. salary')
plt.show()
```



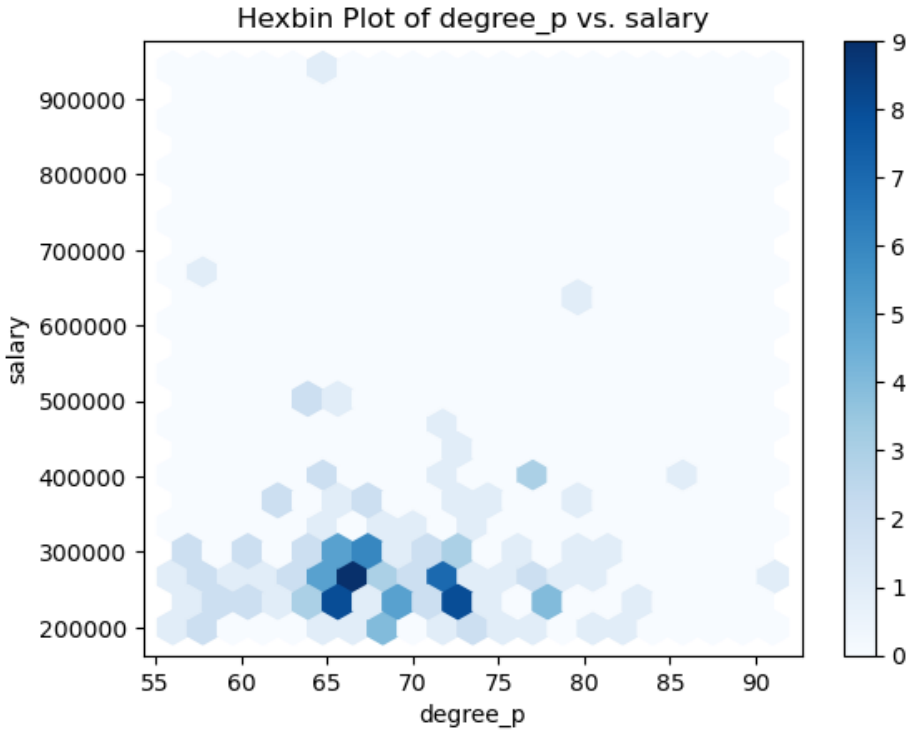
In [ ]:
```

```

degree_p

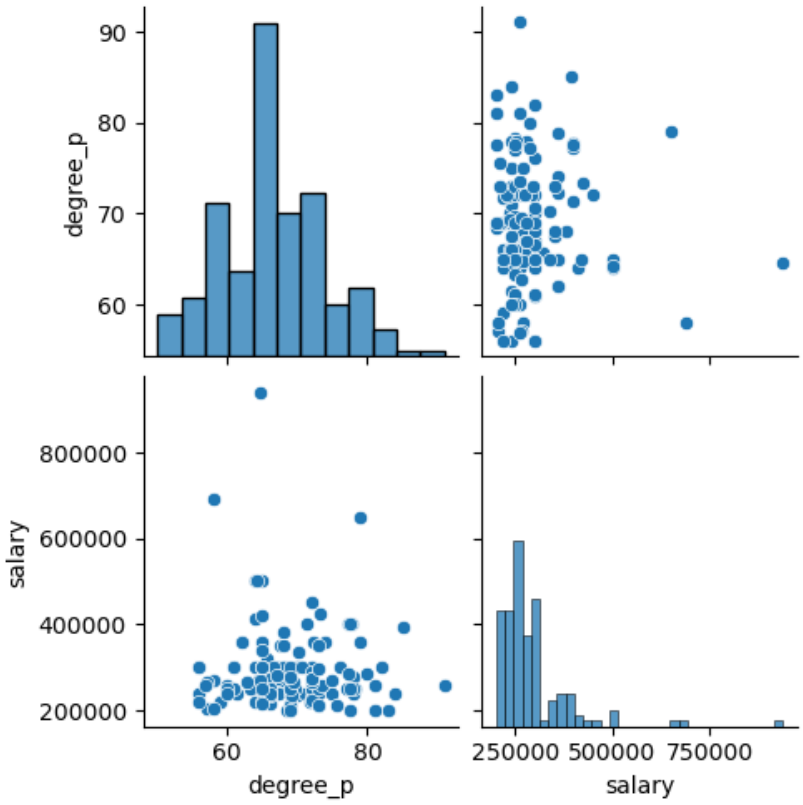
In [75]:
```
plt.scatter(df['degree_p'], df['salary'])
plt.xlabel('degree_p')
plt.ylabel('salary')
plt.title('Scatter Plot of degree_p vs.salary')
plt.show()
```



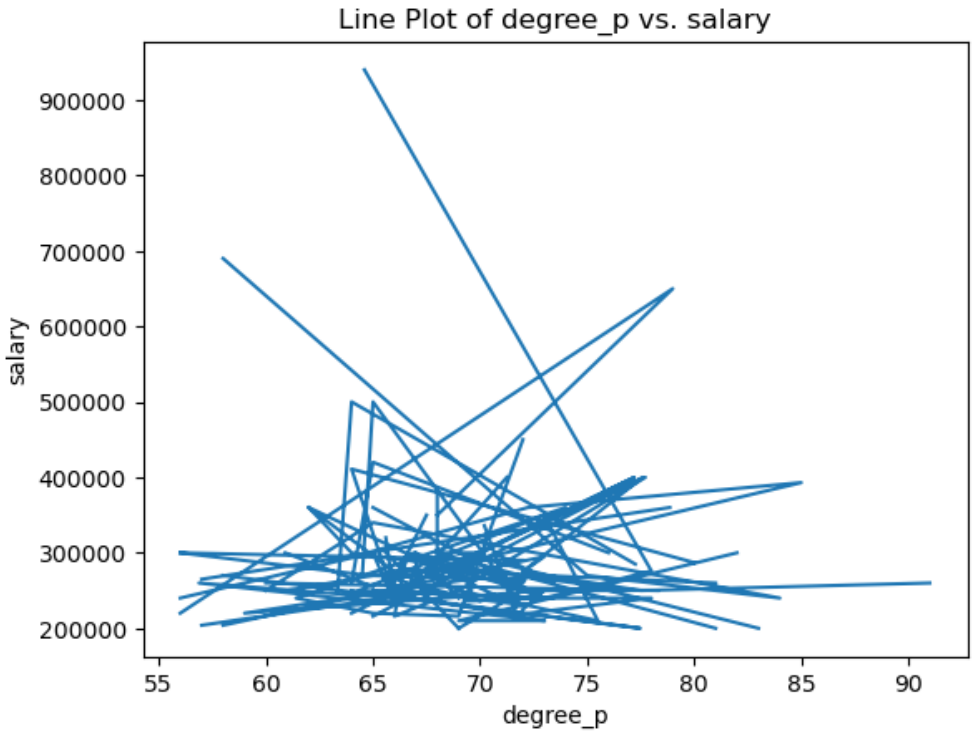
In [76]:
```
plt.hexbin(df['degree_p'], df['salary'], gridsize=20, cmap='Blues')
plt.xlabel('degree_p')
plt.ylabel('salary')
plt.title('Hexbin Plot of degree_p vs. salary')
plt.colorbar()
plt.show()
```



In [77]:
```
sns.pairplot(df[['degree_p', 'salary']])
plt.show()
```

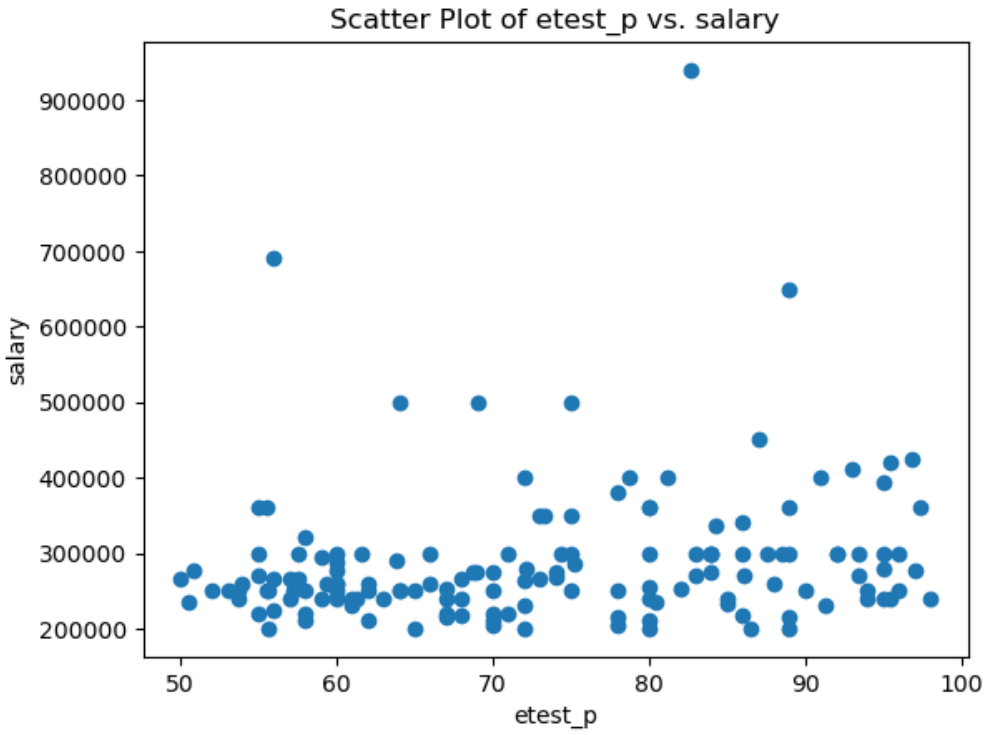


In [78]:
```
plt.plot(df['degree_p'], df['salary'])
plt.xlabel('degree_p')
plt.ylabel('salary')
plt.title('Line Plot of degree_p vs. salary')
plt.show()
```

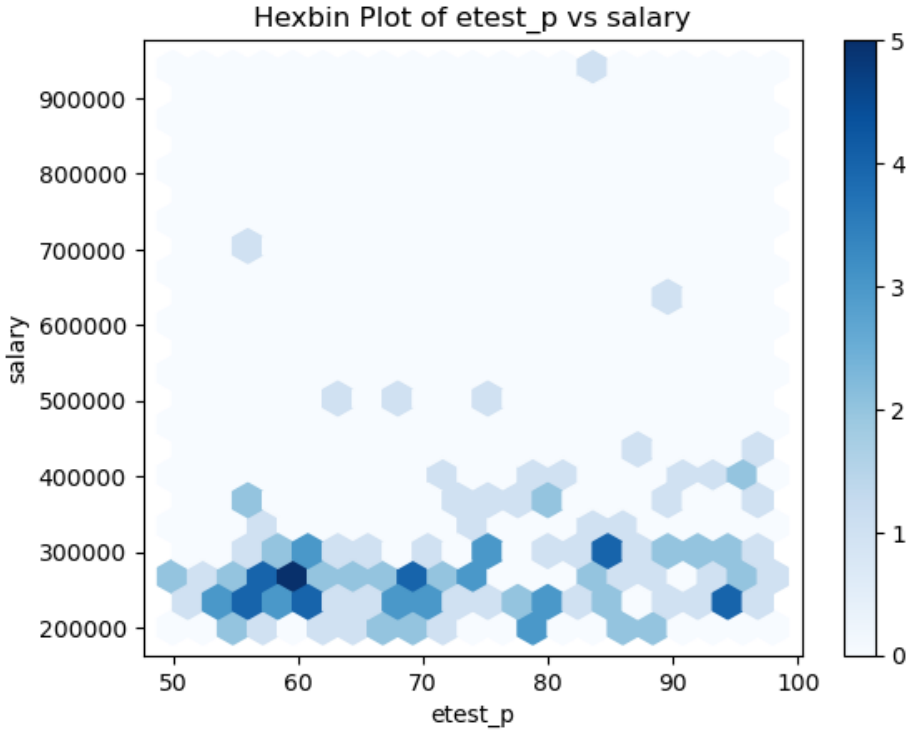


etest_p

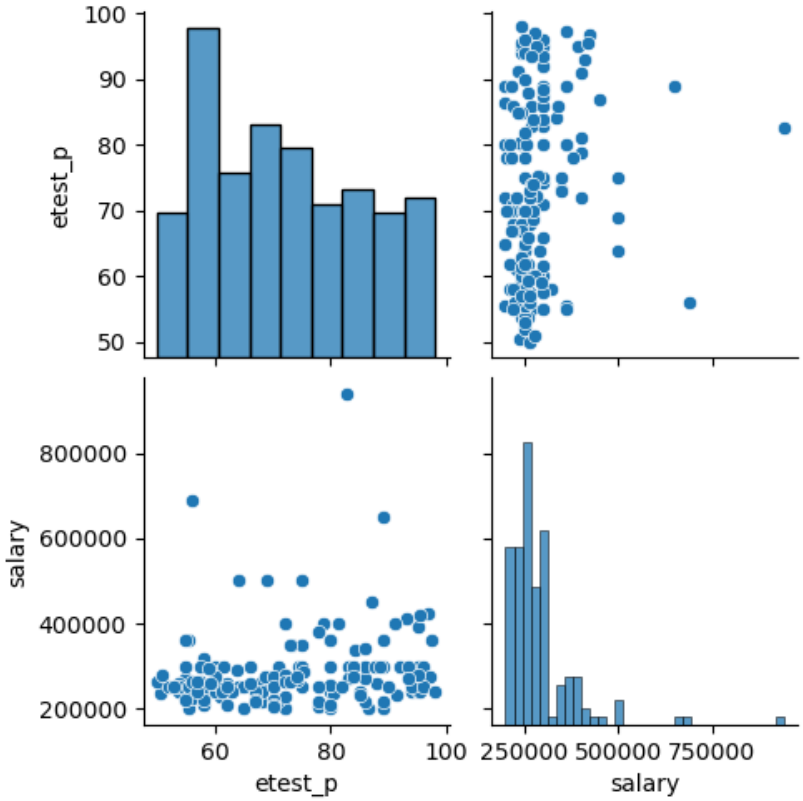
In [79]:
```
plt.scatter(df['etest_p'], df['salary'])
plt.xlabel('etest_p')
plt.ylabel('salary')
plt.title('Scatter Plot of etest_p vs. salary')
plt.show()
```



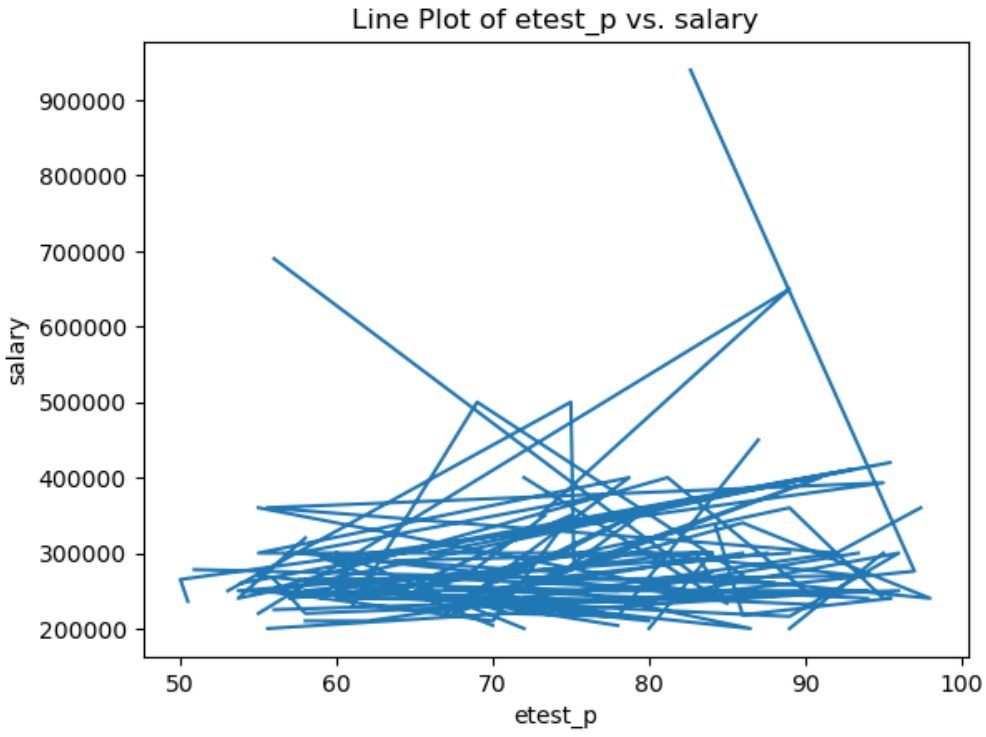
In [80]:
```
plt.hexbin(df['etest_p'], df['salary'], gridsize=20, cmap='Blues')
plt.xlabel('etest_p')
plt.ylabel('salary')
plt.title('Hexbin Plot of etest_p vs salary')
plt.colorbar()
plt.show()
```



In [81]:
```
sns.pairplot(df[['etest_p', 'salary']])
plt.show()
```

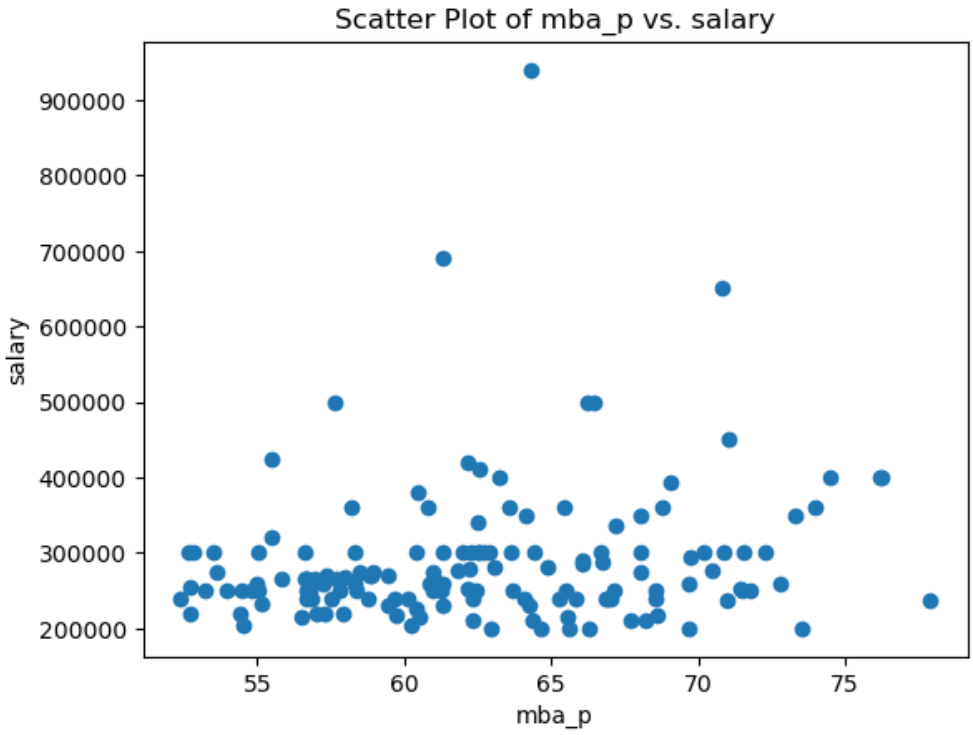


In [82]:
```
plt.plot(df['etest_p'], df['salary'])
plt.xlabel('etest_p')
plt.ylabel('salary')
plt.title('Line Plot of etest_p vs. salary')
plt.show()
```

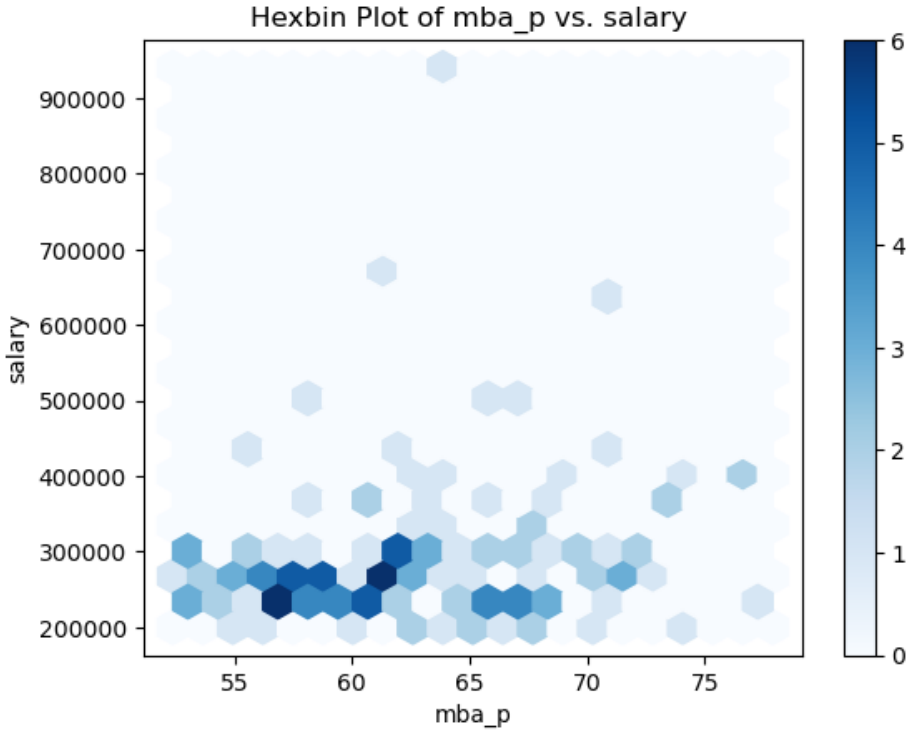


mba_p

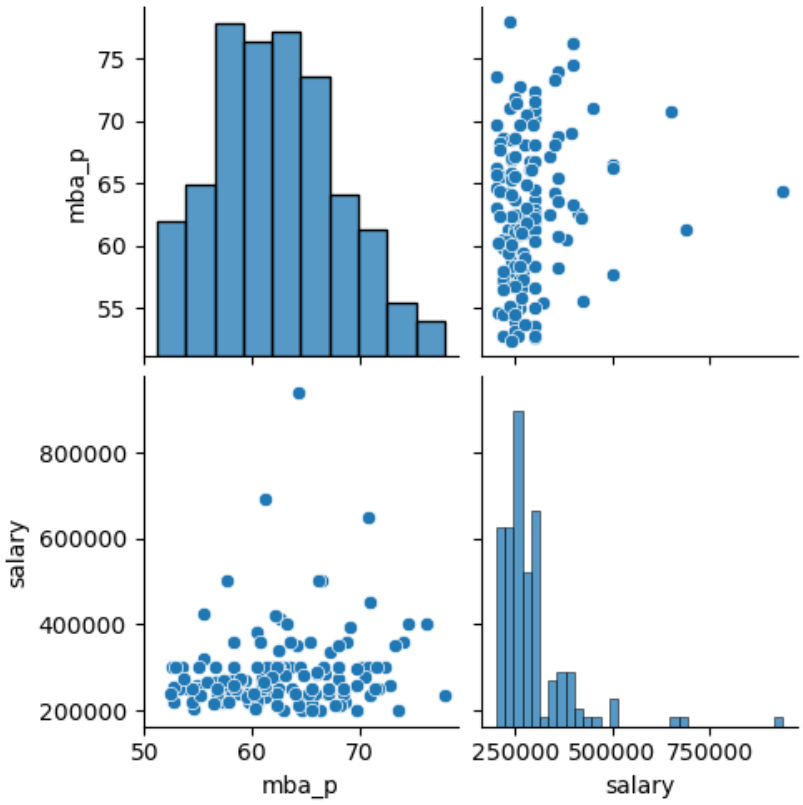
In [83]:
```
plt.scatter(df['mba_p'], df['salary'])
plt.xlabel('mba_p')
plt.ylabel('salary')
plt.title('Scatter Plot of mba_p vs. salary')
plt.show()
```



In [84]:
```
plt.hexbin(df['mba_p'], df['salary'], gridsize=20, cmap='Blues')
plt.xlabel('mba_p')
plt.ylabel('salary')
plt.title('Hexbin Plot of mba_p vs. salary ')
plt.colorbar()
plt.show()
```



In [85]:
```
sns.pairplot(df[['mba_p', 'salary']])
plt.show()
```

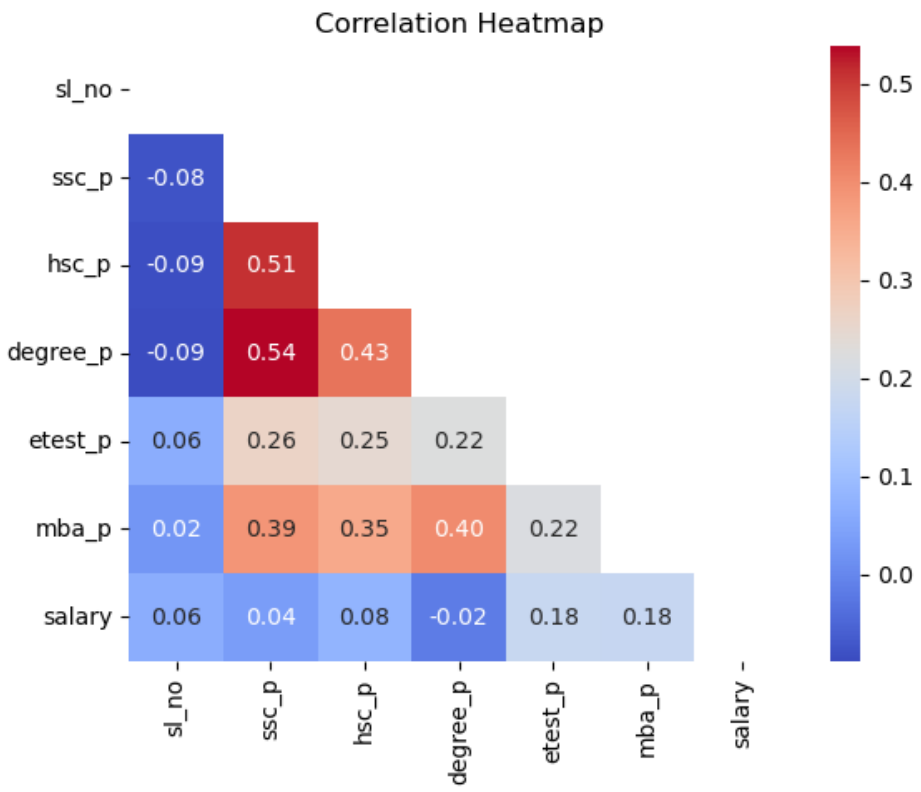


In [86]:
```
plt.plot(df['mba_p'], df['salary'])
plt.xlabel('mba_p')
plt.ylabel('salary')
plt.title('Line Plot of mba_p vs. salary')
plt.show()
```

In [87]:
```
# Assuming 'df' is your DataFrame
columns_to_drop = ['gender','ssc_b','hsc_b','hsc_s','degree_t','workex','specialisation','status'] # Re

# Use the 'drop' method to remove the specified columns
df1 = df.drop(columns=columns_to_drop)
```

In [88]:
```
corr_matrix = df1.corr()
mask = np.triu(corr_matrix)

sns.heatmap(corr_matrix, annot=True, fmt=".2f", cmap="coolwarm", mask=mask)
```

```
plt.title('Correlation Heatmap')
plt.show()
```



In [ ]: