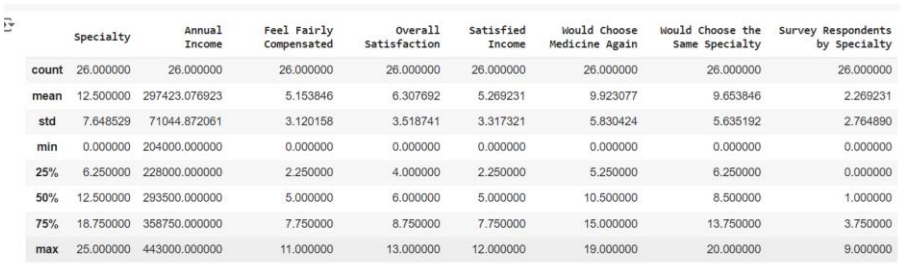
Data Collection and Preprocessing Phase

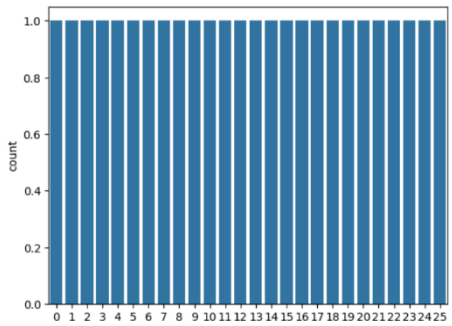
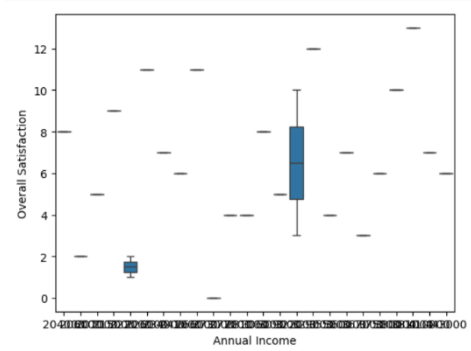
Date	15 July 2024
Team ID	739750
Project Title	Doctors Annual Salary prediction
Maximum Marks	6 Marks

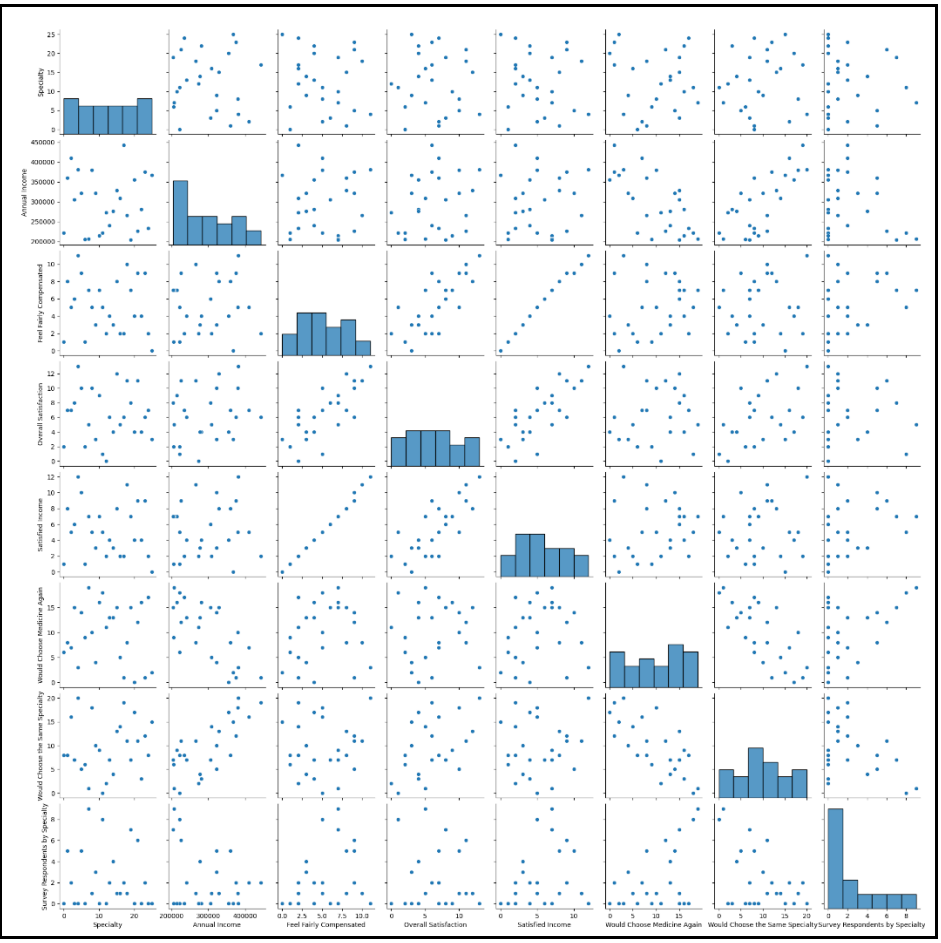
Data Exploration and Preprocessing Report

Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

Section	Description
Data Overview	Dimension: 28 rows × 8 columns Descriptive statistics: 
Countplot Analysis	

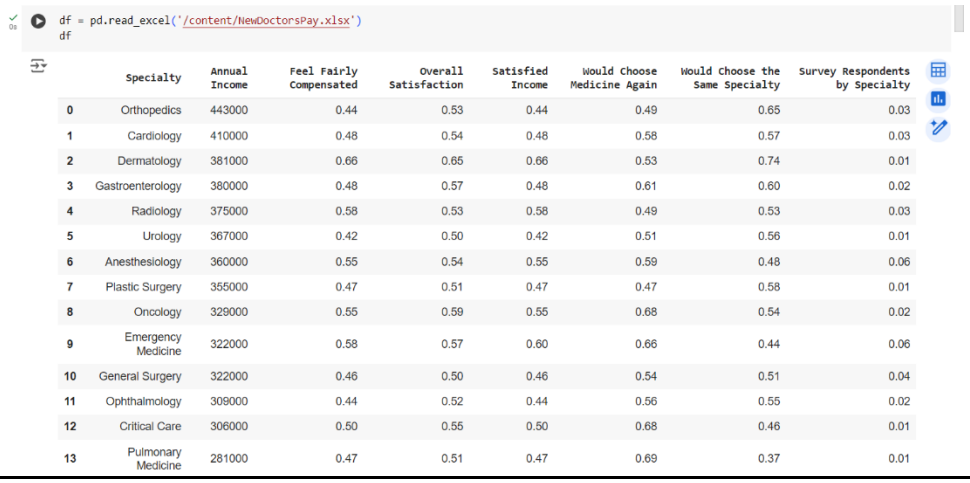
	Specialty	Annual Income	Feel Fairly Compensated	Overall Satisfaction	Satisfied Income	Would Choose Medicine Again	Would Choose the Same Specialty	Survey Respondents by Specialty
count	26.000000	26.000000	26.000000	26.000000	26.000000	26.000000	26.000000	26.000000
mean	12.500000	297423.076923	5.153846	6.307692	5.269231	9.923077	9.653846	2.269231
std	7.648529	71044.872061	3.120158	3.518741	3.317321	5.830424	5.635192	2.764890
min	0.000000	204000.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	6.250000	228000.000000	2.250000	4.000000	2.250000	5.250000	6.250000	0.000000
50%	12.500000	293500.000000	5.000000	6.000000	5.000000	10.500000	8.500000	1.000000
75%	18.750000	358750.000000	7.750000	8.750000	7.750000	15.000000	13.750000	3.750000
max	25.000000	443000.000000	11.000000	13.000000	12.000000	19.000000	20.000000	9.000000

	
Boxplot Analysis	

PairPlot Analysis	

Outliers and Anomalies	-

Data Preprocessing Code Screenshots

Loading Data	
Data Transformation	
Feature Engineering	Attached the codes in final submission.
Save Processed Data	-

```
df = pd.read_excel('/content/NewDoctorsPay.xlsx')
df
```

	Specialty	Annual Income	Feel Fairly Compensated	Overall Satisfaction	Satisfied Income	Would Choose Medicine Again	Would Choose the Same Specialty	Survey Respondents by Specialty
0	Orthopedics	443000	0.44	0.53	0.44	0.49	0.65	0.03
1	Cardiology	410000	0.48	0.54	0.48	0.58	0.57	0.03
2	Dermatology	381000	0.66	0.65	0.66	0.53	0.74	0.01
3	Gastroenterology	380000	0.48	0.57	0.48	0.61	0.60	0.02
4	Radiology	375000	0.58	0.53	0.58	0.49	0.53	0.03
5	Urology	367000	0.42	0.50	0.42	0.51	0.56	0.01
6	Anesthesiology	360000	0.55	0.54	0.55	0.59	0.48	0.06
7	Plastic Surgery	355000	0.47	0.51	0.47	0.47	0.58	0.01
8	Oncology	329000	0.55	0.59	0.55	0.68	0.54	0.02
9	Emergency Medicine	322000	0.58	0.57	0.60	0.66	0.44	0.06
10	General Surgery	322000	0.46	0.50	0.46	0.54	0.51	0.04
11	Ophthalmology	309000	0.44	0.52	0.44	0.56	0.55	0.02
12	Critical Care	306000	0.50	0.55	0.50	0.68	0.46	0.01
13	Pulmonary Medicine	281000	0.47	0.51	0.47	0.69	0.37	0.01

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=42)

x_train = x_train.replace('[\$,%]','', regex=True).astype('float') / 100
x_test = x_test.replace('[\$,%]', '', regex=True).astype('float') / 100
y_train = y_train.replace('[\$,%]','', regex=True).astype('float') / 100
y_test = y_test.replace('[\$,%]','', regex=True).astype('float') / 100

imputer_x = SimpleImputer(strategy='mean')
x_train = pd.DataFrame(imputer_x.fit_transform(x_train))
x_test = pd.DataFrame(imputer_x.transform(x_test))

imputer_y = SimpleImputer(strategy='mean')
y_train = imputer_y.fit_transform(y_train.values.reshape(-1, 1))
y_test = imputer_y.transform(y_test.values.reshape(-1, 1))

reg = LinearRegression()
reg.fit(x_train, y_train)
```