Ajeet K. Jain, M. Narsimlu
(ML TEAM)- SONET, KMIT, Hyderabad

Session – 31

This session deals with

Introduction to Case Study

Case Study -01

Problem statement

- Subsidy Inc. delivers subsidies to individuals based on their income
- Accurate income data is one of the hardest piece of data to obtain across the world
- Subsidy Inc. has obtained a large data set of authenticated data on individual income, demographic parameters, and a few financial parameters
- Subsidy Inc. wishes us to :

  Develop an income classifier system for individuals

<u>The Objective is to:</u>

Simplify the data system by reducing the number of variables to be studied, without sacrificing too much of accuracy. Such a system would help Subsidy Inc. in planning subsidy outlay, monitoring and preventing misuse.

Required modules

```
#To visualize the data
import seaborn as sns
#To work with dataframes
import pandas as pd
#To perform numerical operations
import numpy as np
#To partition the data
from sklearn.model_selection import train_test_split
#importing the library for logistic regression
from sklearn.linear_model import LogisticRegression
#importing performance metrics
from sklearn.metrics import accuracy_score,confusion_matrix
```

Data Description

```
#importing data
data_income=pd.read_csv("income.csv")
#create a copy of original data
df_income=data_income.copy()
print(df_income.describe())
```

Numerical Data Description

	age	capitalgain	capitalloss	hoursperweek
count	31978.000000	31978.000000	31978.000000	31978.000000
mean	38.579023	1064.360623	86.739352	40.417850
std	13.662085	7298.596271	401.594301	12.345285
min	17.000000	0.000000	0.000000	1.000000
25%	28.000000	0.000000	0.000000	40.000000
50%	37.000000	0.000000	0.000000	40.000000
75%	48.000000	0.000000	0.000000	45.000000
max	90.000000	99999.000000	4356.000000	99.000000

Information about Data Set

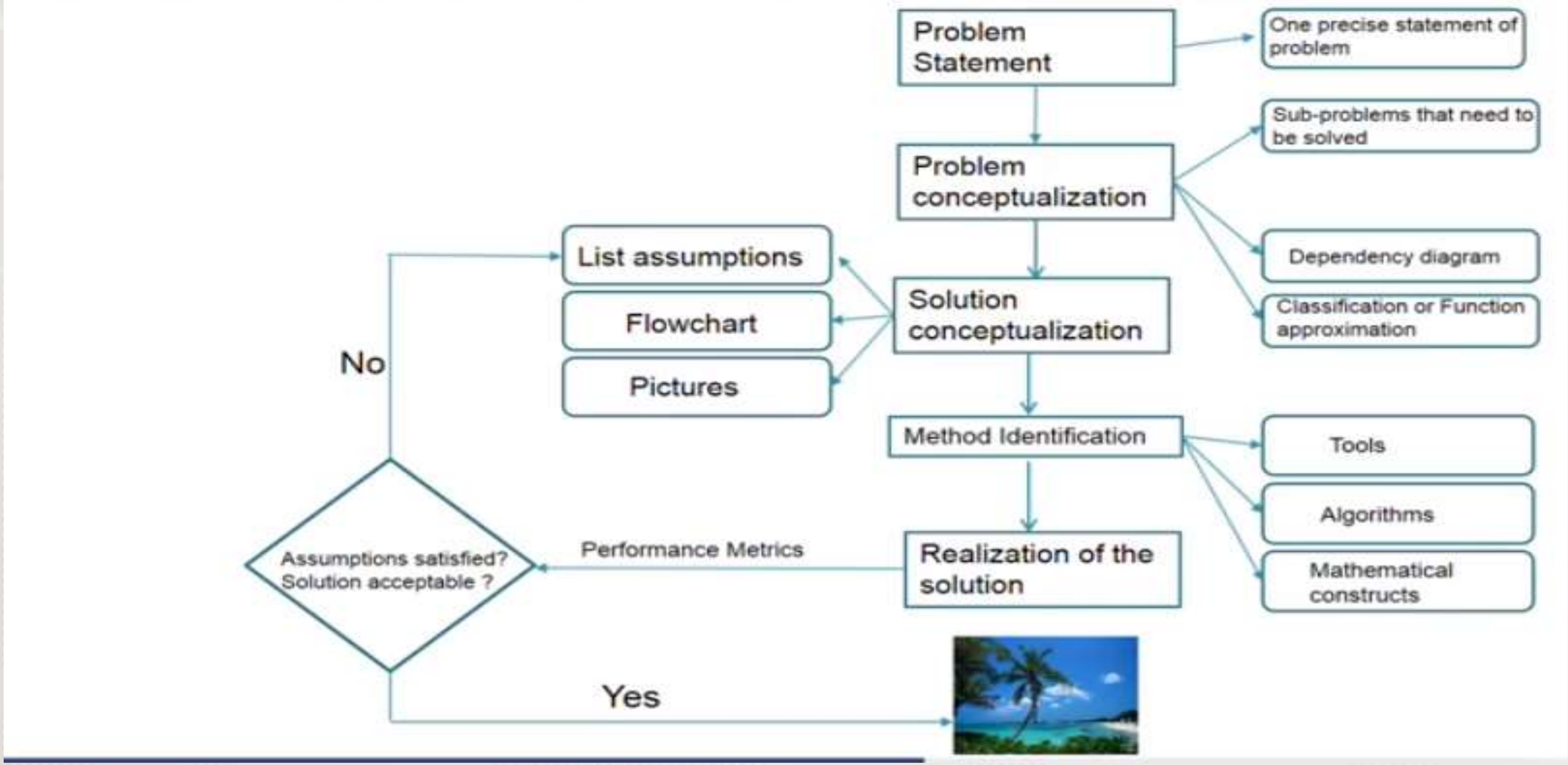
```
#importing data
data_income=pd.read_csv("income.csv")
#create a copy of original data
df_income=data_income.copy()
print(df_income.info())
```

Output

```
RangeIndex: 31978 entries, 0 to 31977
Data columns (total 13 columns):
age                 31978 non-null int64
JobType             31978 non-null object
EdType              31978 non-null object
maritalstatus       31978 non-null object
occupation          31978 non-null object
relationship        31978 non-null object
race                31978 non-null object
gender              31978 non-null object
capitalgain         31978 non-null int64
capitalloss         31978 non-null int64
hoursperweek        31978 non-null int64
nativecountry       31978 non-null object
SalStat             31978 non-null object
```

Data Analytics Frame work

Data analytics framework



Problem conceptualization

Framework

- Problem conceptualization
  - Develop an income classifier for individuals with reduced no. of variables
- Problem characterization- Classification

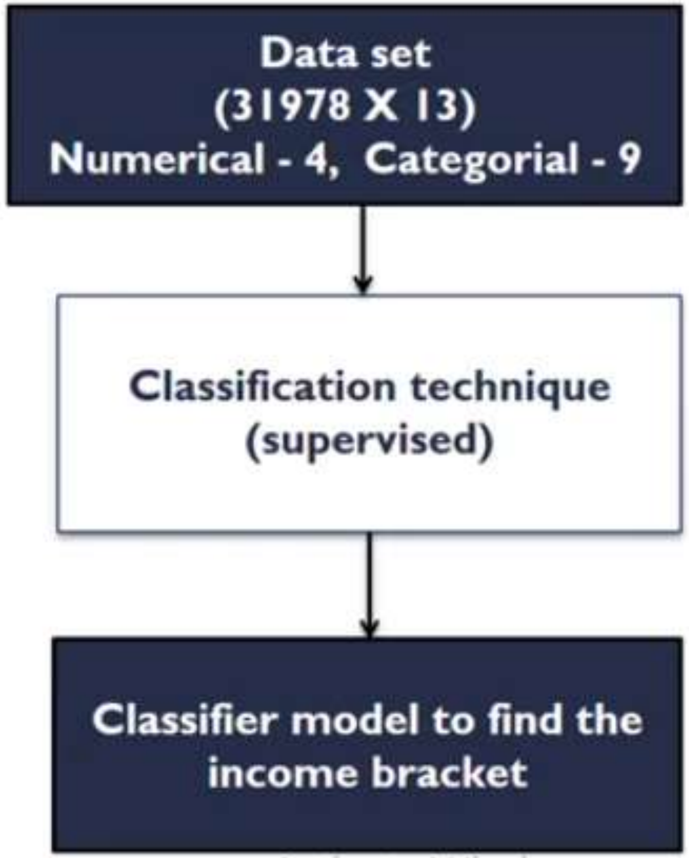
Apriori Known:

✓ Dependent variable

categorical (binary)

✓ Independent variables

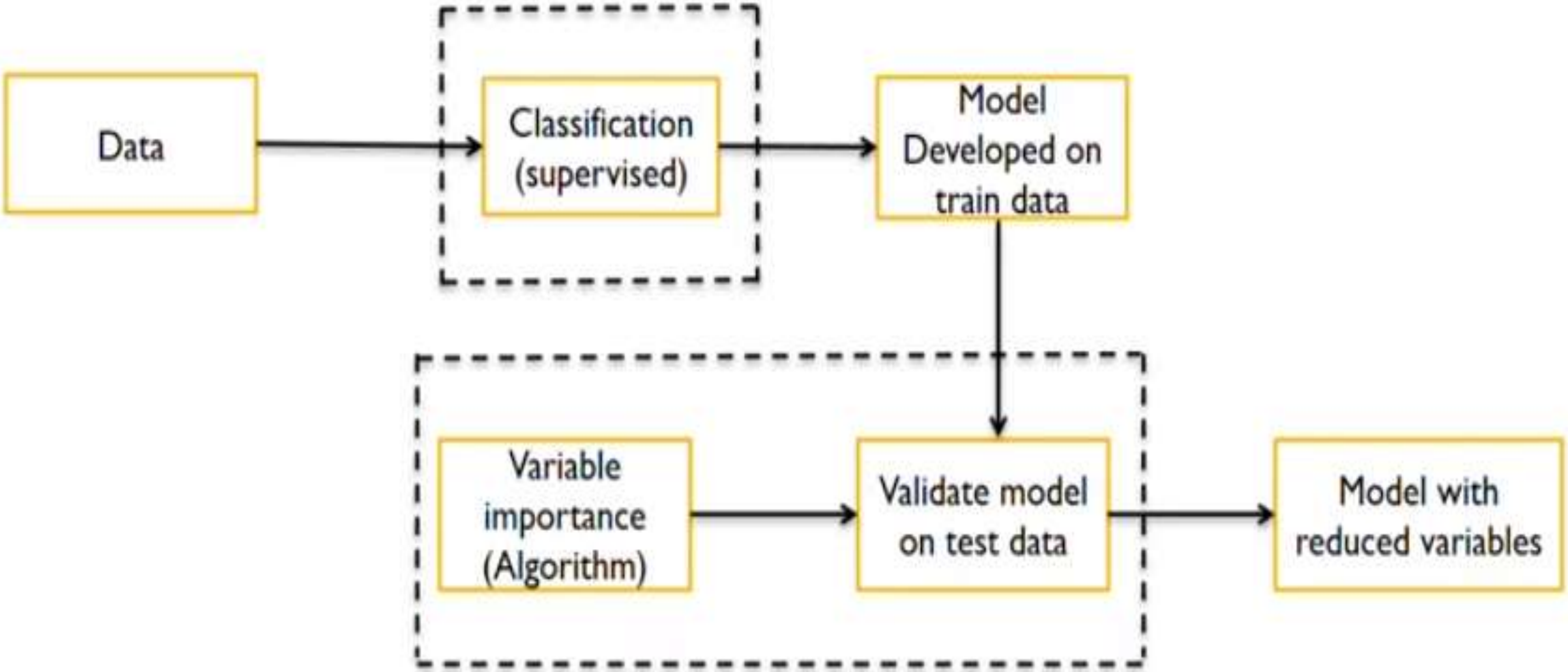
numerical + categorical



Framework

Framework

- Flow chart:



Solution conceptualization

Framework

- Solution conceptualization
  - Identify if data is clean
  - Look for missing values
  - Identify variables influencing salary status and look for possible relationships between variables
    - Correlation, chi-square test, box plots, scatter plots etc.
  - Identify if categories can be combined
  - Build a model with reduced number of variables to classify the individual's salary status to plan subsidy outlay, monitor and pr
misuse

Method Identification

Framework

- Method identification
  - Logistic Regression
  - Random Forest
  - K Nearest Neighbors
- Realization of solution
  - Evaluate performance metrics
  - If assumptions are satisfied and solutions are acceptable then model is

Data Exploratory Analysis

DATA SCIENCE

```
"""
EDA
1.getting to know the data
2.Data Preprocessin
3.Cross tables and data visualization
"""
#1.Getting to know the data
#To check variables data type
print(df_income.info())
#To find the missing values in each feature
print(df_income.isnull().sum())
#No missing values
#Summary of numerical variables
print(df_income.describe())
#Summary of categorical variables
print(df_income.describe(include="O"))
#Frequency of each categories
print(df_income["JobType"].value_counts())
```

Data Exploratory

```
#importing data
df_income=pd.read_csv("income.csv")
#checking for unique classes
print(np.unique((df_income["JobType"])))
print(np.unique(df_income["occupation"]))
#checking other special characters in the dataset
data=pd.read_csv("income.csv",na_values=[" ?"])
#Check missing values in each feature
print(data.isnull().sum())
missing=data[data.isnull().any(axis=1)]
```

#To consider one missing column
print(missing)

Data Exploratory Analysis

```
Private              22286
Self-emp-not-inc      2499
Local-gov             2067
?                     1809
State-gov             1279
Self-emp-inc          1074
Federal-gov            943
Without-pay             14
Never-worked             7
Name: JobType, dtype: int64
```

```
Prof-specialty       4038
Craft-repair         4030
Exec-managerial      3992
Adm-clerical         3721
Sales                3584
Other-service        3212
Machine-op-inspct    1966
?                    1816
Transport-moving     1572
Handlers-cleaners    1350
Farming-fishing       989
Tech-support          912
Protective-serv       644
Priv-house-serv       143
Armed-Forces            9
```

Finding missing Values and Correlation

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
"""
1.missing values in jobtype -     1809
2.missing values in occupation -  1816
3.There are 1809 rows where tow soecific columns
i.e oocupation and jobtype have missing values
4.(1816-1809)=7 => still we have occupation unfilled for these rows.
because,jobtype is "never worked"
"""
data2=data.dropna(axis=0)
#correlation between independent varaibles
corr_rel=data2.corr()
print(corr_rel)
```

Output

Week4')

	age	capitalgain	capitalloss	hoursperweek
age	1.000000	0.080154	0.060165	0.101599
capitalgain	0.080154	1.000000	-0.032229	0.080432
capitalloss	0.060165	-0.032229	1.000000	0.052417
hoursperweek	0.101599	0.080432	0.052417	1.000000

Age Vs Salary status -EDA

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
#cross tables and data visualization
#extract the columns
print(data2.columns)
gender_tab=pd.crosstab(index=data2["gender"],columns="counts",
                       normalize=True)
print(gender_tab)

#Relation between Gender vs salary
gender_salstat=pd.crosstab(index=data2["gender"],
                           columns=data2["SalStat"],
                           margins=True,normalize="index")
print(gender_salstat)
```

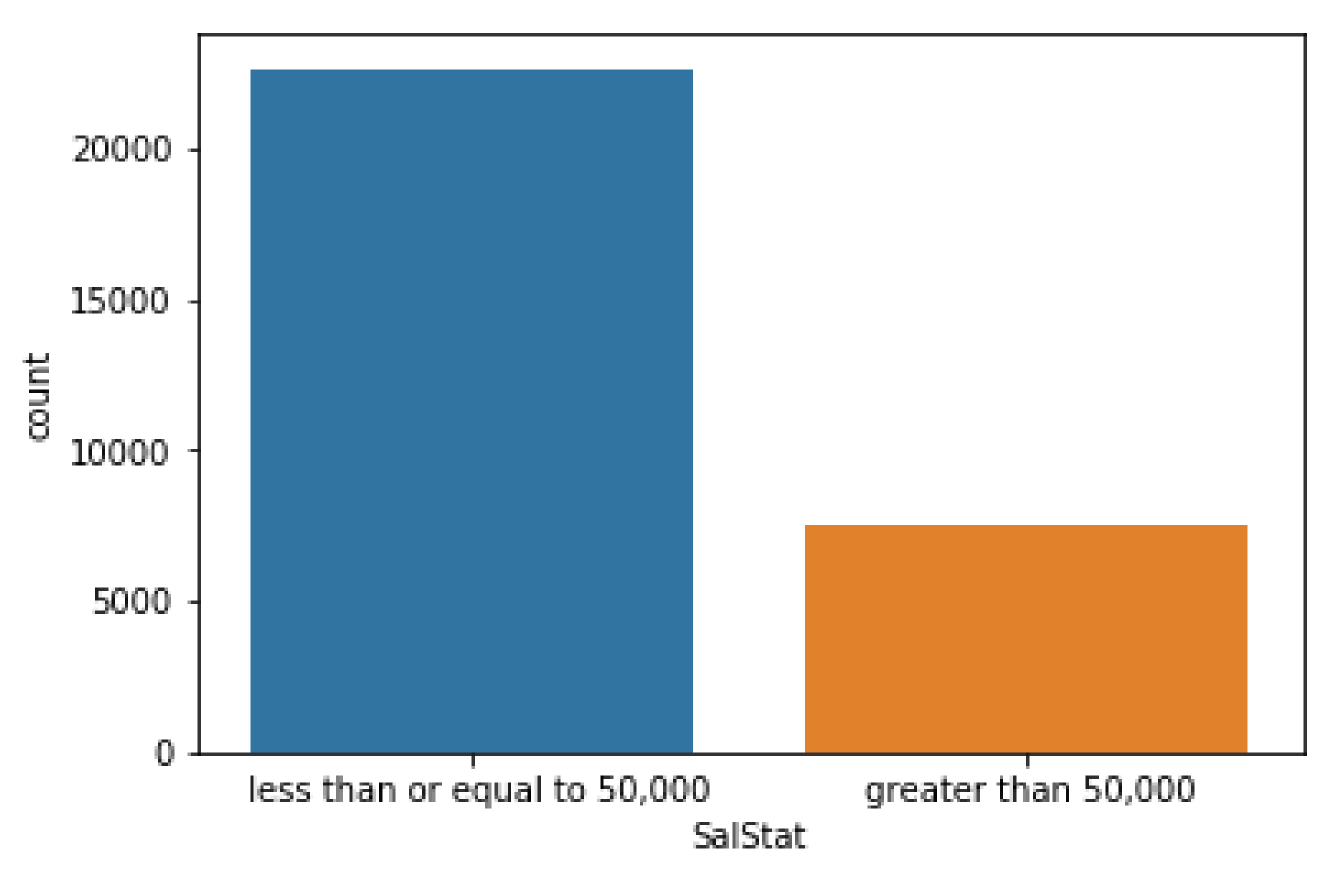
Output

```
Index(['age', 'JobType', 'EdType', 'maritalstatus', 'occupation',
       'relationship', 'race', 'gender', 'capitalgain', 'capitalloss',
       'hoursperweek', 'nativecountry', 'SalStat'],
      dtype='object')
col_0      counts
gender
 Female  0.324315
 Male    0.675685
SalStat   greater than 50,000   less than or equal to 50,000
gender
 Female              0.113678                       0.886322
 Male                0.313837                       0.686163
All                  0.248922                       0.751078
```

Bar plot b/w Salary status

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
#cross tables and data visualization
#frequency distribution of salary status
SalStat=sns.countplot(data2["SalStat"])
"""
75% of people's salary status is <=50,000
25% of people's salary status is >50,000
"""
```

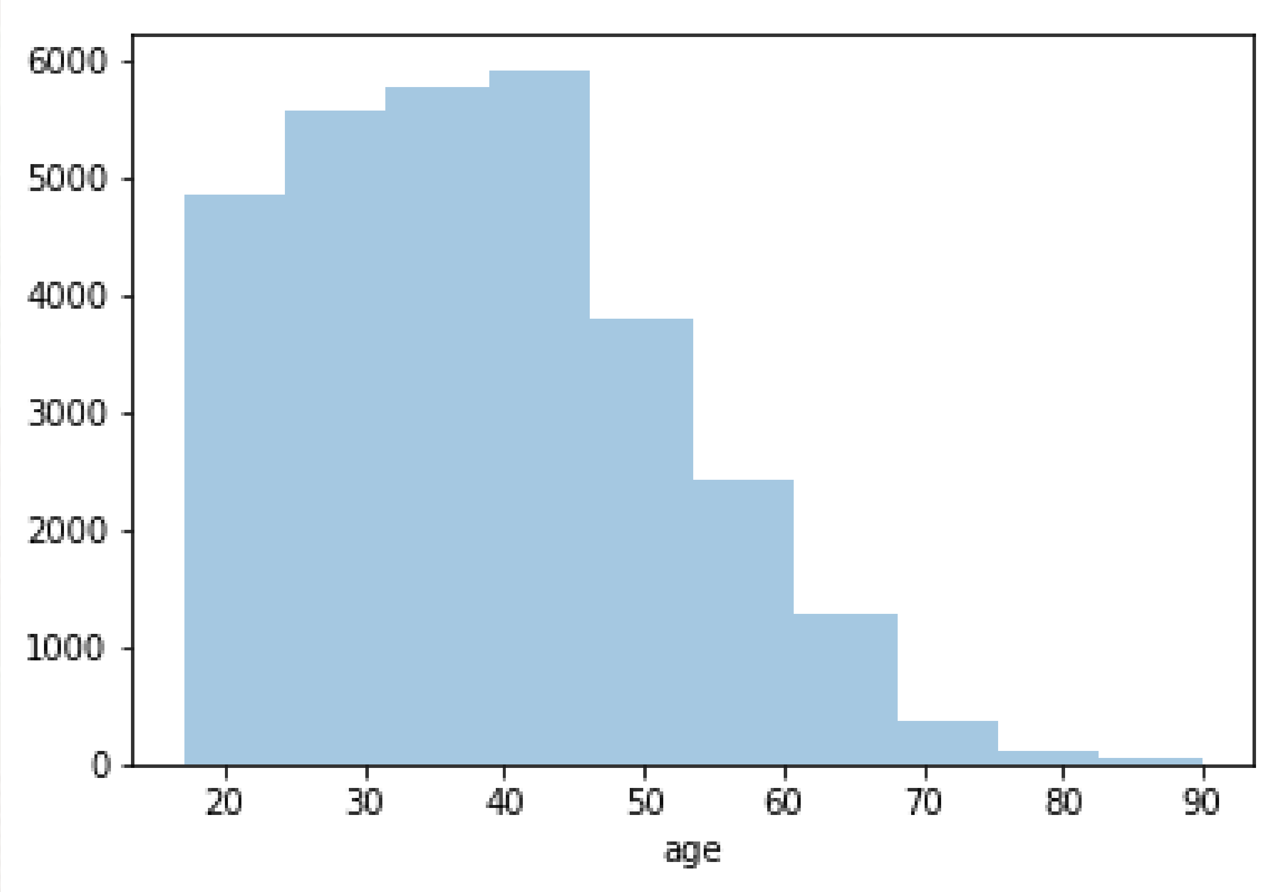
Data Exploratory



Finding Age Frequency

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
#histogram of Age
sns.distplot(data2["age"],bins=10,kde=False)
#people with age 20-45 age are high in frequency
```

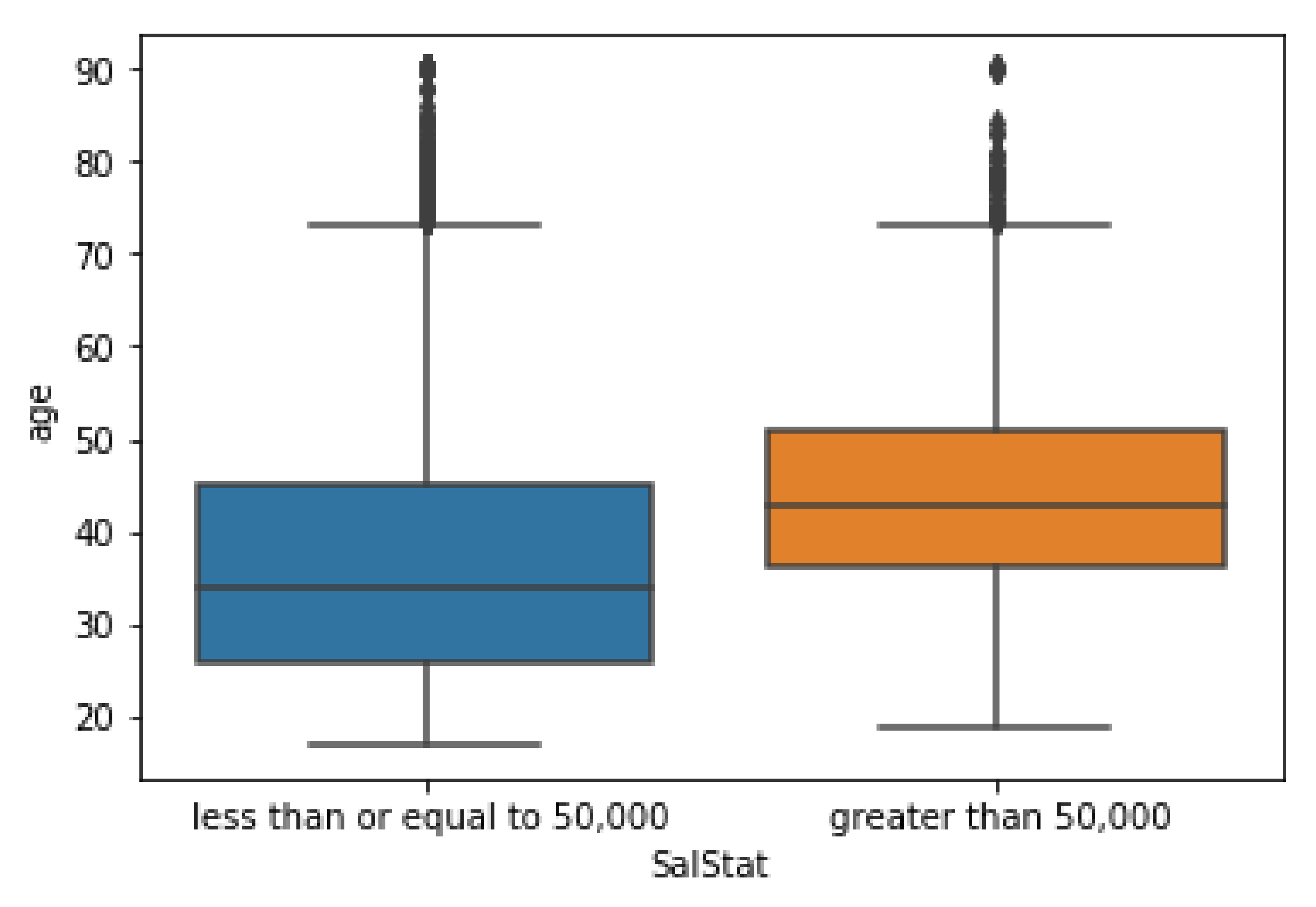
Output



Finding the Outliers

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
#boxplot Age vs salary status
sns.boxplot("SalStat","age",data=data2)
"""

people with 35-50 age are more likely to earn >50000
people with 25-35 age are more likely to earn <=50000
"""
```

Output



90
80
70
60
50
40
30
20
age
less than or equal to 50,000
greater than 50,000
SalStat

SONET

Exploration of JobType vs Salary status

DATA SCIENCE

EDA-

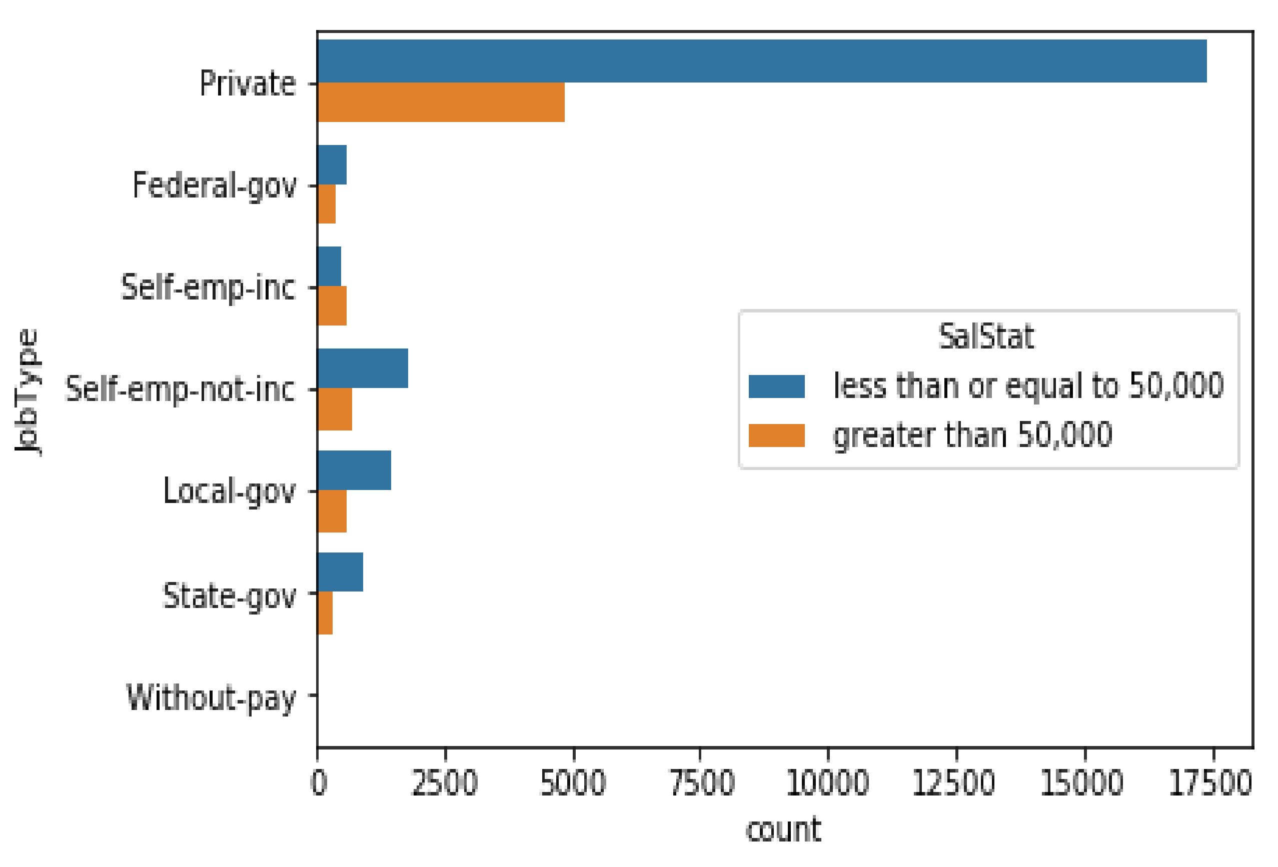
1.Jobtype VS salary status

2.create a cross table Jobtype Vs salary status

```
salStat_tab=pd.crosstab(index=data2["JobType"],
                        columns=data2["SalStat"],
                        margins=True,normalize="index")
print(salStat_tab)
#sns.countplot(y="JobType",data=data2,hue="SalStat")
"""
The above table 56% of self employed
people earn more than 50000 USD per year
"""
```

Output

SalStat	greater than 50,000	less than or equal to 50,000
JobType		
Federal-gov	0.387063	0.612937
Local-gov	0.294630	0.705370
Private	0.218792	0.781208
Self-emp-inc	0.558659	0.441341
Self-emp-not-inc	0.285714	0.714286
State-gov	0.268960	0.731040
Without-pay	0.000000	1.000000
All	0.248922	0.751078



Exercise-1

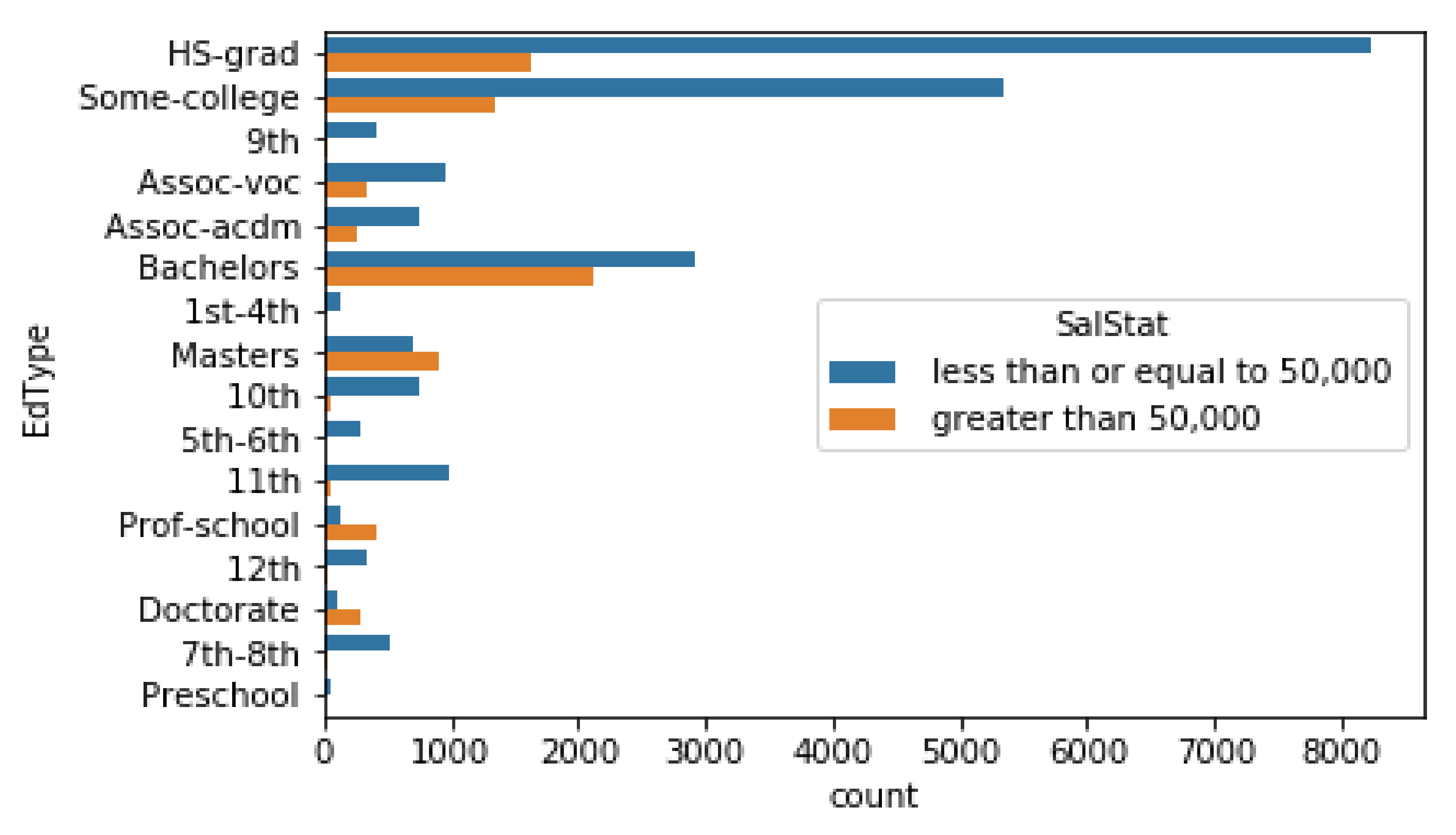
Load the income data set and perform following operations
1.create a bar plot of Education VS salary status
2.create a cross table Education Vs salary status

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
sns.countplot(y="EdType",data=data2,hue="SalStat")
ed_tab=pd.crosstab(index=data2["EdType"],
                        columns=data2["SalStat"],
                        margins=True,normalize="index")
print(ed_tab)
```

The above table we can see that people who have done Doctorate,Masters
,prof-schools are more likely to earn above
50000 USD per year when compared to others
Hence an influencing variable in avoiding the misuse os subsidies

Output

SalStat	greater than 50,000	less than or equal to 50,000
EdType		
10th	0.071951	0.928049
11th	0.056298	0.943702
12th	0.076923	0.923077
1st-4th	0.039735	0.960265
5th-6th	0.041667	0.958333
7th-8th	0.062837	0.937163
9th	0.054945	0.945055
Assoc-acdm	0.253968	0.746032
Assoc-voc	0.263198	0.736802
Bachelors	0.421491	0.578509
Doctorate	0.746667	0.253333
HS-grad	0.164329	0.835671
Masters	0.564229	0.435771
Preschool	0.000000	1.000000
Prof-school	0.749077	0.250923
Some-college	0.200060	0.799940
All	0.248922	0.751078


HS-grad
Some-college
9th
Assoc-voc
Assoc-acdm
Bachelors
1st-4th
Masters
10th
5th-6th
11th
Prof-school
12th
Doctorate
7th-8th
Preschool
EdType
0
1000
2000
3000
4000
5000
6000
7000
8000
count
SalStat
less than or equal to 50,000
greater than 50,000

SONET

Exercise-2

DATA SCIENCE

Load the income data set and perform following operations

1.create a bar plot of Occupation VS salary status

2.create a cross table occupation Vs salary status

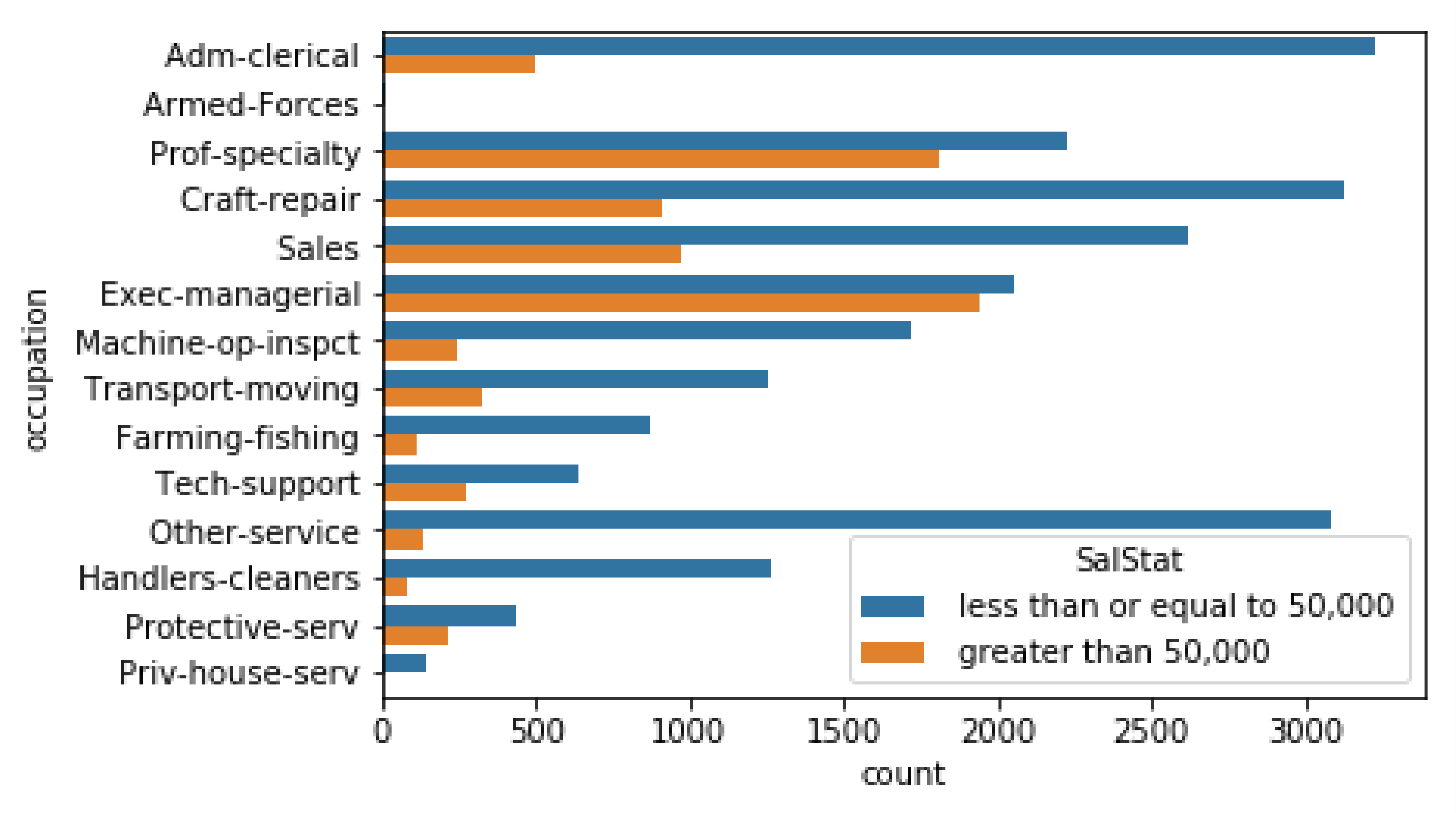
```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
sns.countplot(y="occupation",data=data2,hue="SalStat")
ocup_tab=pd.crosstab(index=data2["occupation"],
                     columns=data2["SalStat"],
                     margins=True,normalize="index")

print(ocup_tab)
```

Output

SalStat	greater than 50,000	less than or equal to 50,000
occupation		
Adm-clerical	0.133835	0.866165
Armed-Forces	0.111111	0.888889
Craft-repair	0.225310	0.774690
Exec-managerial	0.485220	0.514780
Farming-fishing	0.116279	0.883721
Handlers-cleaners	0.061481	0.938519
Machine-op-inspct	0.124619	0.875381
Other-service	0.041096	0.958904
Priv-house-serv	0.006993	0.993007
Prof-specialty	0.448489	0.551511
Protective-serv	0.326087	0.673913
Sales	0.270647	0.729353
Tech-support	0.304825	0.695175
Transport-moving	0.202926	0.797074
All	0.248922	0.751078

Explanation

Those who are making more than 50000 USD per year likely to work
as manager and professional,
hence important variable in avoiding misuse of subsides


Adm-clerical
Armed-Forces
Prof-specialty
Craft-repair
Sales
Exec-managerial
Machine-op-inspct
Transport-moving
Farming-fishing
Tech-support
Other-service
Handlers-cleaners
Protective-serv
Priv-house-serv
occupation
0
500
1000
1500
2000
2500
3000
count
SalStat
less than or equal to 50,000
greater than 50,000

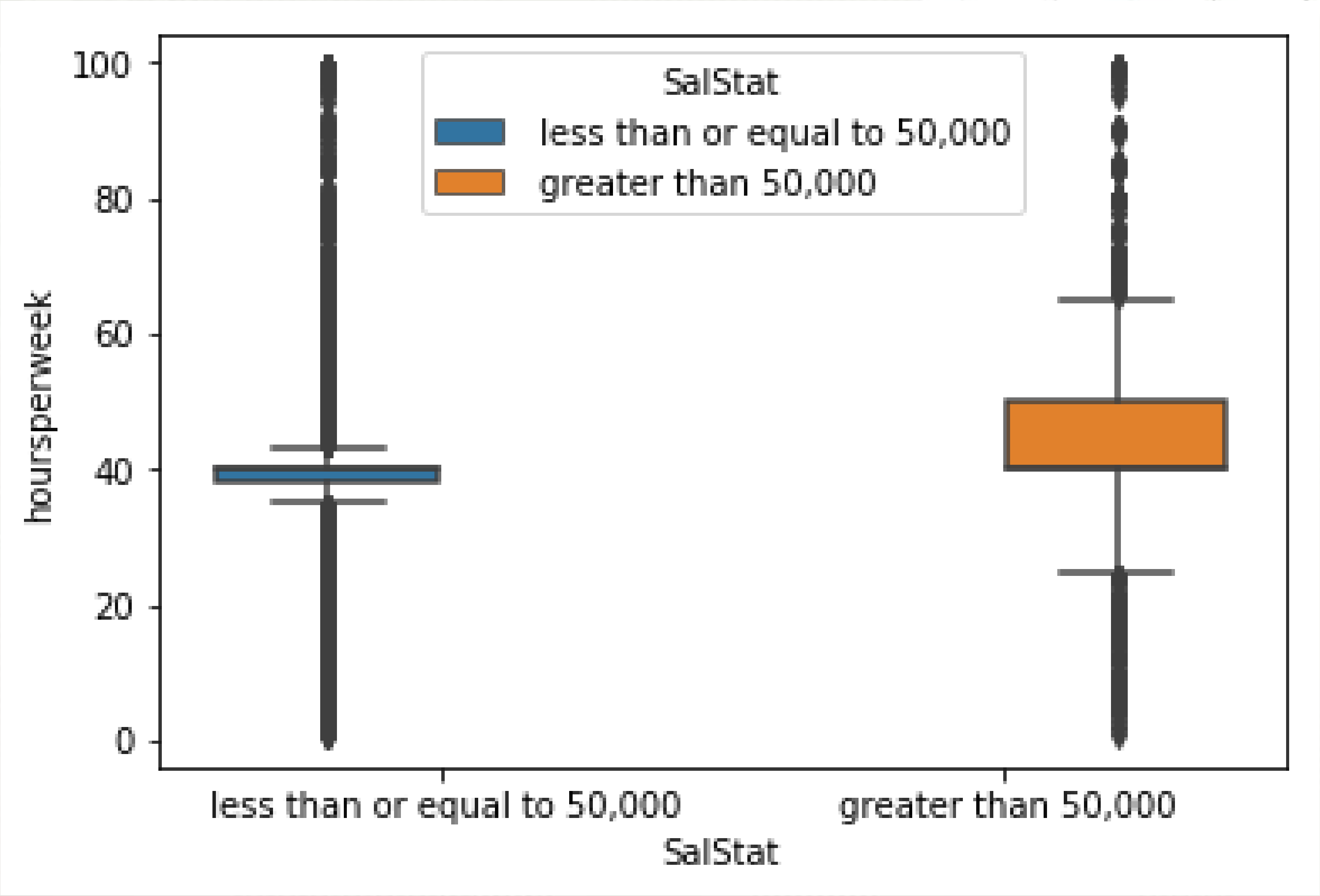
Exercise-3

1.create a boxplot hourperweek Vs salary status

```
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
sns.boxplot(x=data2["SalStat"],
            y=data2["hoursperweek"],
            data=data2,hue="SalStat")
hrsweek_tab=pd.crosstab(index=data2["hoursperweek"],
                        columns=data2["SalStat"],
                        margins=True,normalize="index")
print(hrsweek_tab.head(10))
```

Output

SalStat	greater than 50,000	less than or equal to 50,000
hoursperweek		
1	0.142857	0.857143
2	0.133333	0.866667
3	0.041667	0.958333
4	0.074074	0.925926
5	0.157895	0.842105
6	0.100000	0.900000
7	0.105263	0.894737
8	0.058824	0.941176
9	0.058824	0.941176
10	0.058559	0.941441


SalStat
less than or equal to 50,000
greater than 50,000
100
80
60
40
20
0
hoursperweek
less than or equal to 50,000
greater than 50,000
SalStat

Data Encoding

```
#Logistict Regression
"""
Data encoding
the salary status categories are encoded as 0,1
"""
#importing data
data=pd.read_csv("income.csv",na_values=[" ?"])
data2=data.dropna(axis=0)
#create a cross table on salstatus
print(data2["SalStat"].value_counts())
cat_salStat={" less than or equal to 50,000":0," greater than 50,000":1}
data2["SalStat"].replace(cat_salStat,inplace=True)
print(data2["SalStat"].head(6))
new_data=pd.get_dummies(data2,drop_first=True)
#storing the column names
column_list=list(new_data.columns)
print(column_list)
```

Identifying input features

```
#seperating input features from the data
features=list(set(column_list)-set(["SalStat"]))
print(features)
#Storing output variable values in y
y=new_data["SalStat"].values
print(y)
#storing input feature values in x
x=new_data[features].values
print(x)
```

Splitting Data and Model fitting

```
#splitting the data into train and test
train_x,test_x,train_y,test_y=train_test_split(x,y,test_size=0.3,
                                               random_state=0)
#creating a instance of logistict regression
logistic=LogisticRegression()
#fitting the values for x and y
logistic.fit(train_x,train_y)
#To display the fitting function attributes such as coef,intercept etc..
print(logistic.coef_)
print(logistic.intercept_)
```

Evaluating the model

```
#prediction from the test data
prediction=logistic.predict(test_x)
print(prediction)
#model evolution using classification metrics
#confusion metrics - To display correctly classified data
#and wrongly classified data
confus_matrix=confusion_matrix(test_y,prediction)
print(confus_matrix)
#Calculate the accuracy
accu_score=accuracy_score(test_y,prediction)
print(accu_score)
#To display missclassified values from the prediction
print("Missclassified")
print((test_y!=prediction).sum())
```

Output

```
[0 0 0 ... 0 0 0]
[[6338  485]
 [ 941 1285]]
0.8424135263565035
Missclassified
1426
```

Conclusion

You are aware of

Data Encoding

Project Life Cycle

We will proceed with

Case Study

THANK YOU