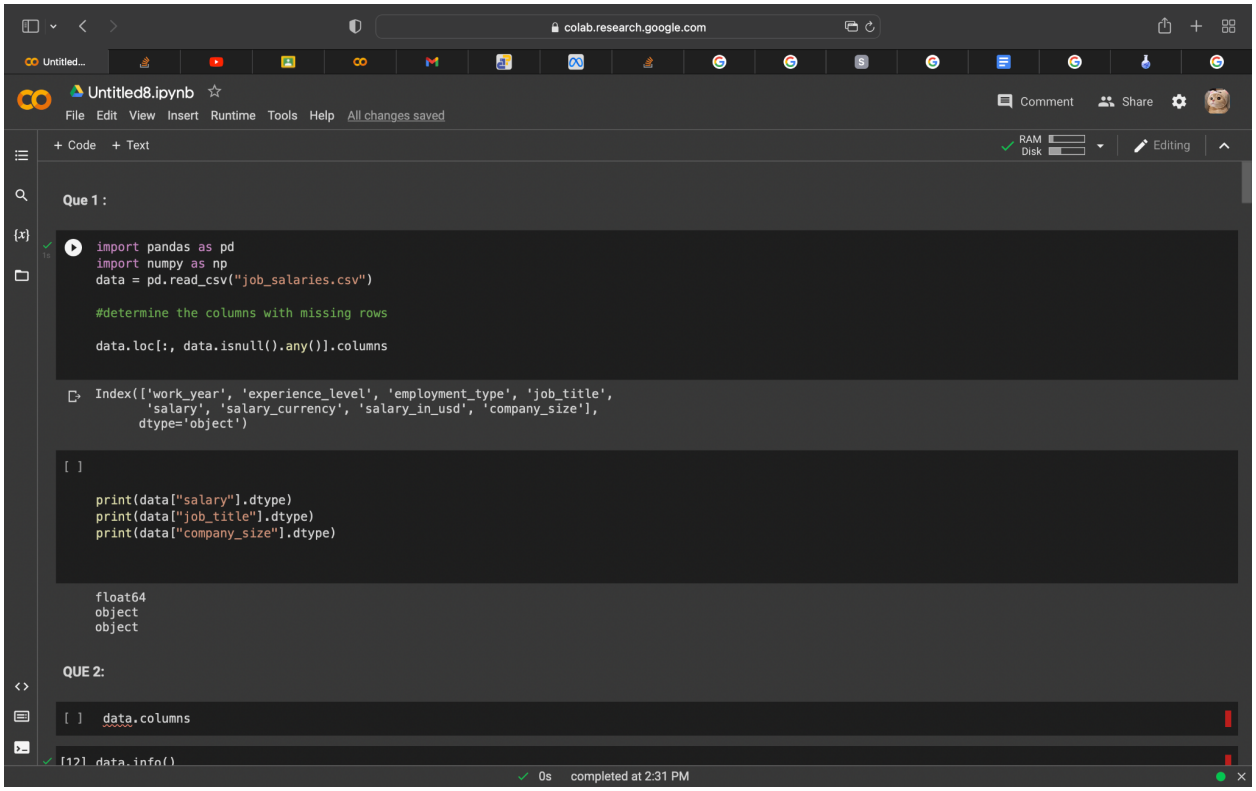
Smruti Dawale
B21BB007
LAB 2

Que 1)

By observation we can tell the data type of the given categories

a) job _title = object
b) Salary = float
c) Company_size =object

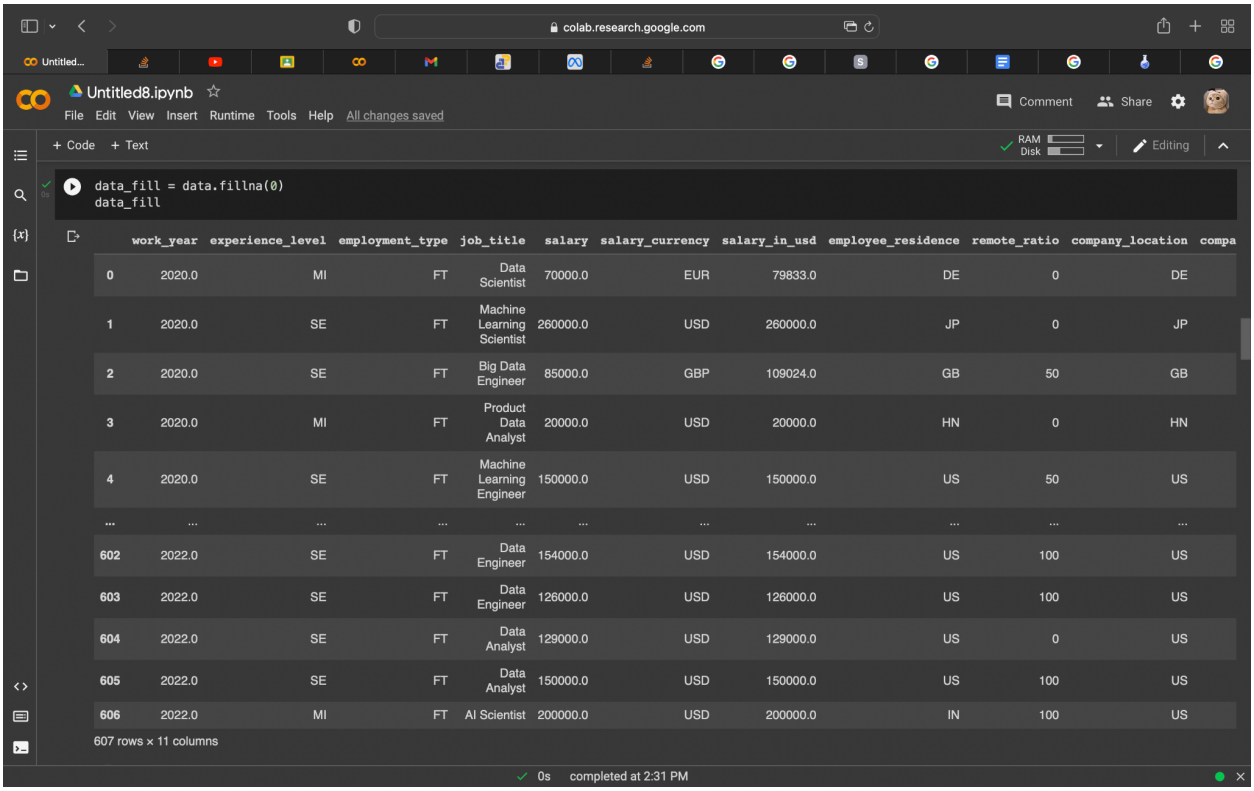
We can verify this by using the code line filename[name_of_column].dtype

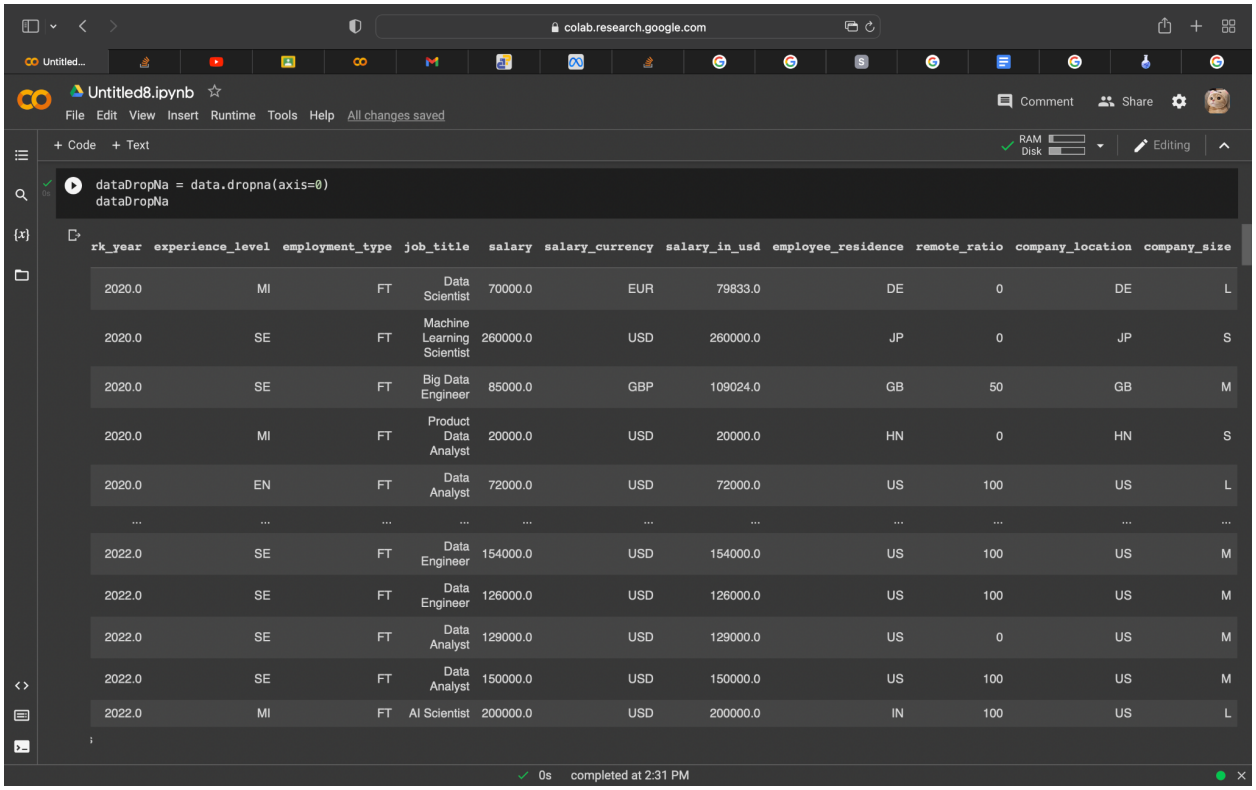


Que 2)

Almost all the algorithms in scikit learn are not capable of handling missing data , so before training your model you must handle your missing data.
One of the ways is “list wise deletion of cases” : discarding observations where value in any of the variables is missing .Here observations are your rows and variables are columns , you basically discard the row if any of the values in your column is missing.
I used `dataDropNa = data.dropna(axis=0)`, thereby you can see 497 rows x 12 columns , instead of the original data where we had 607 rows x 12 columns.

We can also replace all the NA / NaN values by specific value (I used 0) `data_fill = data.fillna(0)`, this basically replaces all of your missing values by 0.



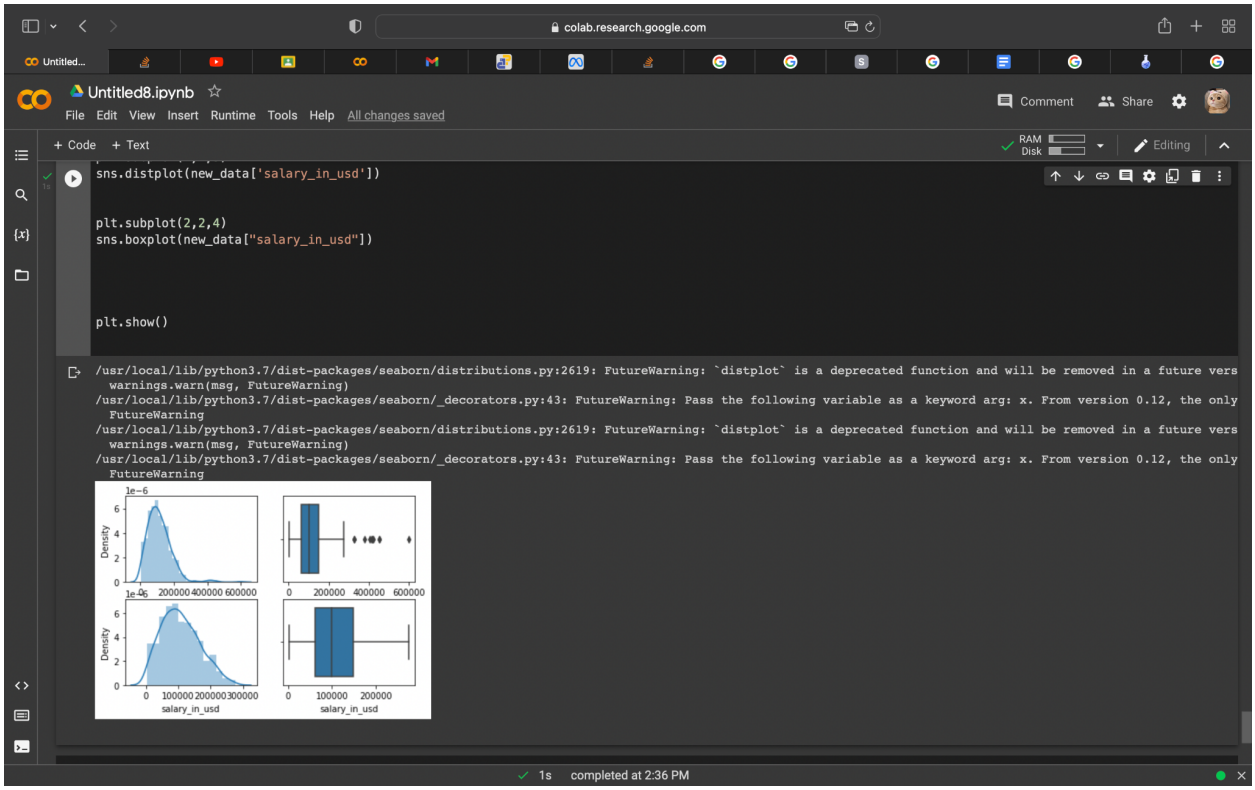


rk_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
2020.0	MI	FT	Data Scientist	70000.0	EUR	79833.0	DE	0	DE	L
2020.0	SE	FT	Machine Learning Scientist	260000.0	USD	260000.0	JP	0	JP	S
2020.0	SE	FT	Big Data Engineer	85000.0	GBP	109024.0	GB	50	GB	M
2020.0	MI	FT	Product Data Analyst	20000.0	USD	20000.0	HN	0	HN	S
2020.0	EN	FT	Data Analyst	72000.0	USD	72000.0	US	100	US	L
...	...	...	...	...	...	...	...	...	...	...
2022.0	SE	FT	Data Engineer	154000.0	USD	154000.0	US	100	US	M
2022.0	SE	FT	Data Engineer	126000.0	USD	126000.0	US	100	US	M
2022.0	SE	FT	Data Analyst	129000.0	USD	129000.0	US	0	US	M
2022.0	SE	FT	Data Analyst	150000.0	USD	150000.0	US	100	US	M
2022.0	MI	FT	AI Scientist	200000.0	USD	200000.0	IN	100	US	L

Que 3)

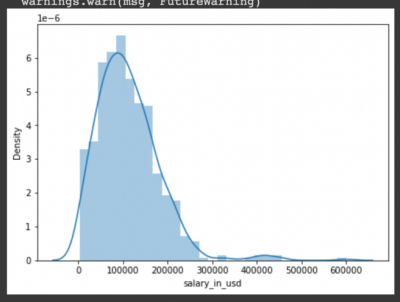
You can reduce noise in various ways , one of them is removing the outliers.Outliers are basically those **data points that are significantly different from the rest of the dataset**. They are often abnormal observations that skew the data distribution, and arise due to inconsistent data entry, or erroneous observations.

I have used the IQR method , since the data(i.e the salary_in_usd) is skewed in nature .
We define a Interquartile Range which is (75 - 25)percentile and whiskers using the formula.all the points lying outside the whiskers are our outliers.
Here in the histogram plot you can observe our data is right skewed (skewness can be determined using (`data["salary_in_usd"].skew()`) In the box plot you can see the outliers on the right side (`1.664144558187621`).now we define a new data set whose value should be less than the upper limit . And basically after plotting the new data you can see that the outliers got removed .


colab.research.google.com
Untitled8.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
Editing
sns.distplot(new_data['salary_in_usd'])
plt.subplot(2,2,4)
sns.boxplot(new_data["salary_in_usd"])
plt.show()
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future vers
warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only
FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future vers
warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only
FutureWarning
1e−6
Density
salary_in_usd
1s completed at 2:36 PM

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(16,5))
plt.subplot(1,2,1)
sns.distplot(data["salary_in_usd"])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future vers
  warnings.warn(msg, FutureWarning)
```



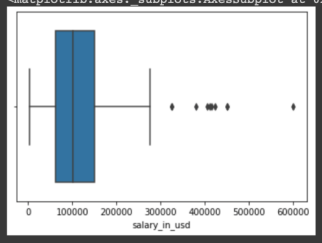

```
data["salary_in_usd"].skew()
```

```
data["salary_in_usd"].describe()
```

0s completed at 2:35 PM

```
sns.boxplot(data["salary_in_usd"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f956a441750>
```




```
percentile25 = data["salary_in_usd"].quantile(0.25)
percentile75 = data["salary_in_usd"].quantile(0.75)
```

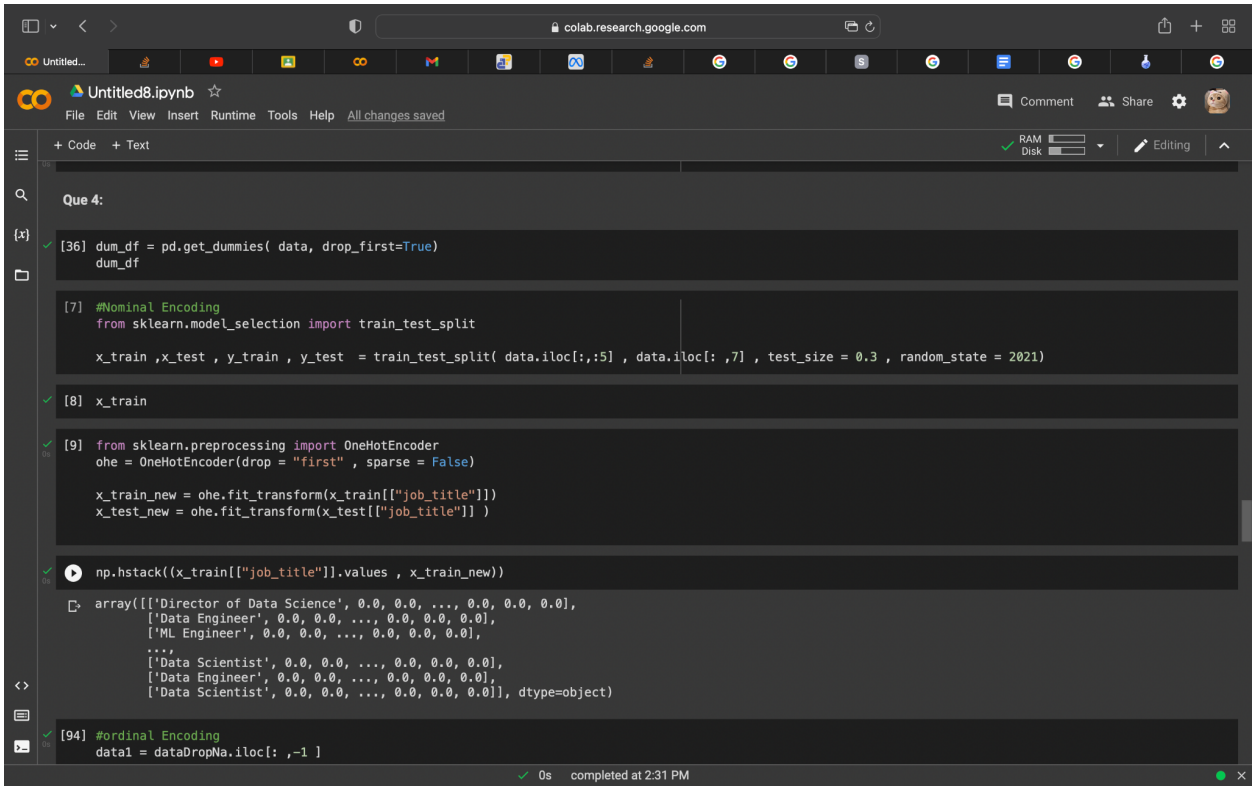
```
iqr = percentile75 - percentile25
iqr
```

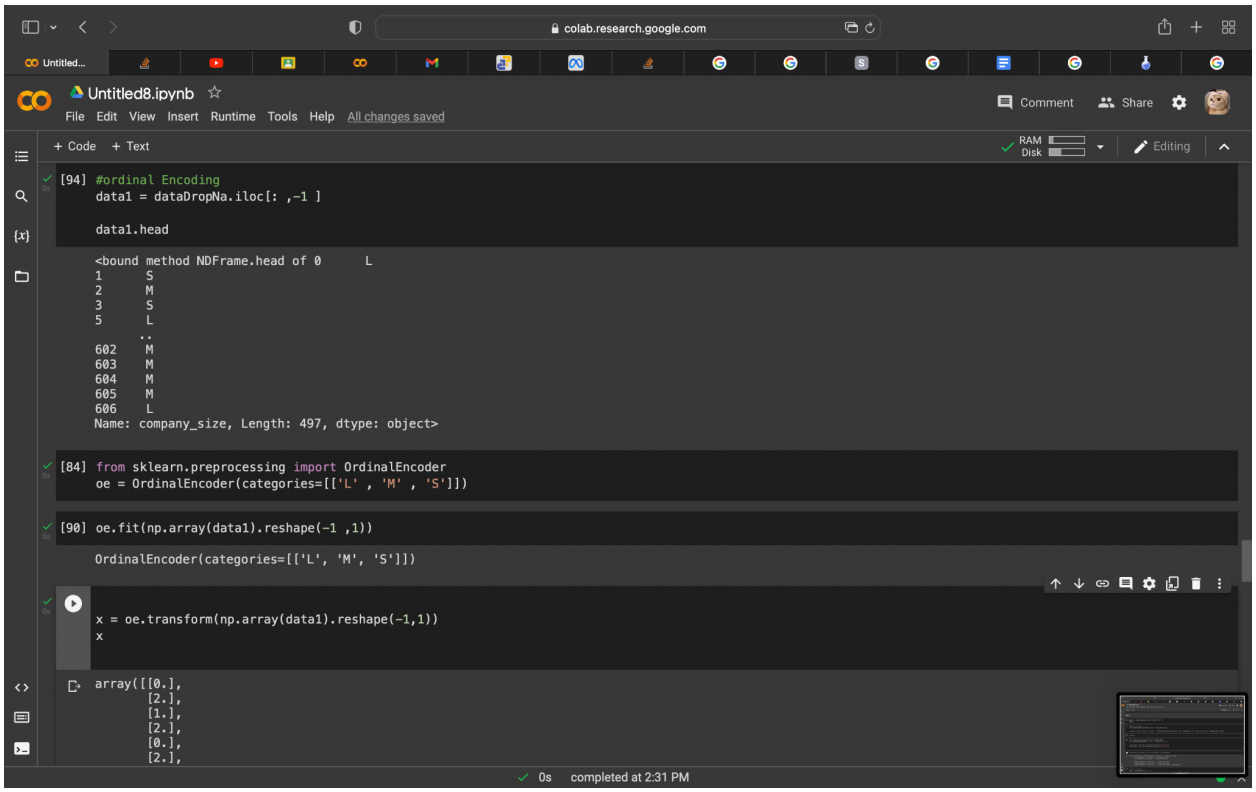
```
upperlimit = percentile75 + 1.5*iqr
lowerlimit = percentile25 - 1.5*iqr

print(upperlimit , lowerlimit)
```

0s completed at 2:35 PM

Que 4)

Categorical data can be divided into 2 types Nominal data (here you cannot define a relationship between the data) and Ordinal data (here you can define a relationship between the data).Categorical data is present mostly in the form of strings and machine learning algorithms expect numbers , so it's your duty to convert these categories to numbers .there are various type of encoding , one is Ordinal encoding which is used for the Ordinal type of data and OneHotEncoding is used for Nominal data .
Firstly i splitted the data into test and train proving the job_title as x and y as salary_in_usd
The OneHotEncoder converts the data into arrays by using the hstack i stacked the x_train and the previous data column.

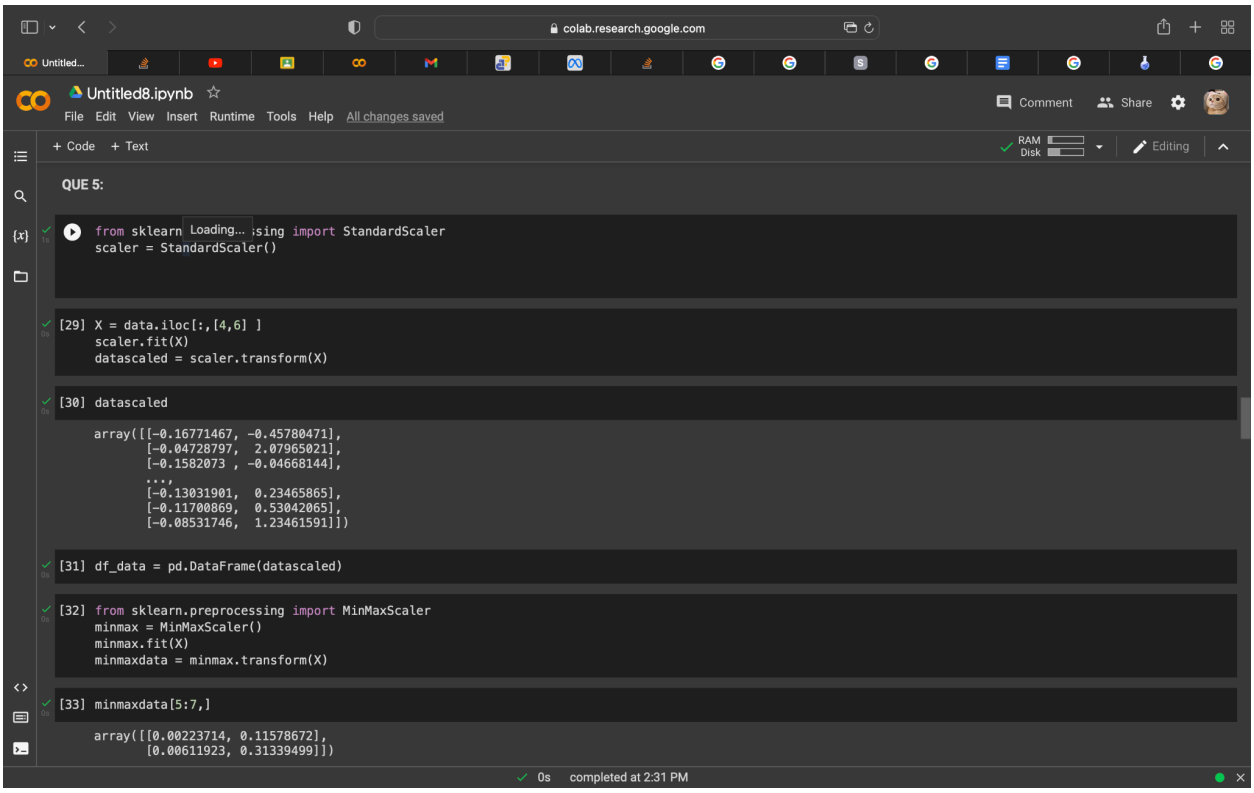

colab.research.google.com
Untitled8.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Que 4:
[36] dum_df = pd.get_dummies(data, drop_first=True)
dum_df
[7] #Nominal Encoding
from sklearn.model_selection import train_test_split
x_train ,x_test , y_train , y_test = train_test_split(data.iloc[:,:5] , data.iloc[: ,7] , test_size = 0.3 , random_state = 2021)
[8] x_train
[9] from sklearn.preprocessing import OneHotEncoder
ohe = OneHotEncoder(drop = "first" , sparse = False)
x_train_new = ohe.fit_transform(x_train[["job_title"]])
x_test_new = ohe.fit_transform(x_test[["job_title"]])
np.hstack((x_train[["job_title"]].values , x_train_new))
array([['Director of Data Science', 0.0, 0.0, ..., 0.0, 0.0, 0.0],
['Data Engineer', 0.0, 0.0, ..., 0.0, 0.0, 0.0],
['ML Engineer', 0.0, 0.0, ..., 0.0, 0.0, 0.0],
...,
['Data Scientist', 0.0, 0.0, ..., 0.0, 0.0, 0.0],
['Data Engineer', 0.0, 0.0, ..., 0.0, 0.0, 0.0],
['Data Scientist', 0.0, 0.0, ..., 0.0, 0.0, 0.0]], dtype=object)
[94] #ordinal Encoding
data1 = dataDropNa.iloc[: ,-1]
0s completed at 2:31 PM


colab.research.google.com
Untitled8.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[94] #ordinal Encoding
data1 = dataDropNa.iloc[: ,-1]
data1.head
<bound method NDFrame.head of 0 L
1 S
2 M
3 S
5 L
..
602 M
603 M
604 M
605 M
606 L
Name: company_size, Length: 497, dtype: object>
[84] from sklearn.preprocessing import OrdinalEncoder
oe = OrdinalEncoder(categories=[['L' , 'M' , 'S']])
[90] oe.fit(np.array(data1).reshape(-1 ,1))
OrdinalEncoder(categories=[['L', 'M', 'S']])
x = oe.transform(np.array(data1).reshape(-1,1))
x
array([[0.],
[2.],
[1.],
[2.],
[0.],
[2.],
0s completed at 2:31 PM

Que 5)

You do scaling of the data to minimize the errors , making the numerical data relatable so the computer can minimize the errors.

Minmax basically converts the data into the range 0 to 1 .



Que 6)
b)

Train test split , here you basically split the data into 2 sets , where you use the training set for the training purpose rather than training the whole data.

For dividing the data into 3 parts , I first put the train and validation set together (main)and then the splitted the main into train and validation.