dmv3

November 3, 2024

1 Data Cleaning and Preparation Problem Statement: Analyzing Customer Churn in a Telecommunications Company The goal is to perform data cleaning and preparation to gain insights into the factors that contribute to customer churn

[23]:
```
import numpy as  np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler,LabelEncoder
from sklearn.model_selection import train_test_split
```

[24]:
```
df=pd.read_csv(r"C:\Users\dell\Desktop\DMV and ML\DMV Datasets\telecom_churn.
↪csv")
```

[25]:
```
df.head()
```

[25]:

	customer_id	telecom_partner	gender	age	state	city	pincode	date_of_registration	num_dependents	estimated_salary	calls_made	sms_sent	data_used	churn
0	1	Reliance Jio	F	25	Karnataka	Kolkata	755597	2020-01-01	4	124962	44	45	-361	0
1	2	Reliance Jio	F	55	Mizoram	Mumbai	125926	2020-01-01	2	130556	62	39	5973	0
2	3	Vodafone	F	57	Arunachal Pradesh	Delhi	423976	2020-01-01	0	148828	49	24	193	1
3	4	BSNL	M	46	Tamil Nadu	Kolkata	522841	2020-01-01	1	38722	80	25	9377	1
4	5	BSNL	F	26	Tripura	Delhi	740247	2020-01-01	2	55098	78	15	1393	0

[26]:
```
df.describe()
```

[26]:

	customer_id	age	pincode	num_dependents
count	243553.000000	243553.000000	243553.000000	243553.000000
mean	121777.000000	46.077609	549501.270541	1.997500
std	70307.839393	16.444029	259808.860574	1.414941
min	1.000000	18.000000	100006.000000	0.000000
25%	60889.000000	32.000000	324586.000000	1.000000
50%	121777.000000	46.000000	548112.000000	2.000000
75%	182665.000000	60.000000	774994.000000	3.000000
max	243553.000000	74.000000	999987.000000	4.000000

	estimated_salary	calls_made	sms_sent	data_used
count	243553.000000	243553.000000	243553.000000	243553.000000
mean	85021.137839	49.010548	23.945404	4993.186025
std	37508.963233	29.453556	14.733575	2942.019547
min	20000.000000	-10.000000	-5.000000	-987.000000
25%	52585.000000	24.000000	11.000000	2490.000000
50%	84990.000000	49.000000	24.000000	4987.000000
75%	117488.000000	74.000000	36.000000	7493.000000
max	149999.000000	108.000000	53.000000	10991.000000

	churn
count	243553.000000
mean	0.200478
std	0.400359
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

[27]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 243553 entries, 0 to 243552
Data columns (total 14 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   customer_id           243553 non-null  int64
 1   telecom_partner       243553 non-null  object
 2   gender                243553 non-null  object
 3   age                   243553 non-null  int64
 4   state                 243553 non-null  object
 5   city                  243553 non-null  object
 6   pincode               243553 non-null  int64
 7   date_of_registration  243553 non-null  object
```

```
 8   num_dependents        243553 non-null  int64
 9   estimated_salary      243553 non-null  int64
 10  calls_made            243553 non-null  int64
 11  sms_sent              243553 non-null  int64
 12  data_used             243553 non-null  int64
 13  churn                 243553 non-null  int64
dtypes: int64(9), object(5)
memory usage: 26.0+ MB
```

[28]:
```
df.shape
```

[28]:
```
(243553, 14)
```

[29]:
```
df.isna().sum()
```

[29]:
```
customer_id             0
telecom_partner         0
gender                  0
age                     0
state                   0
city                    0
pincode                 0
date_of_registration    0
num_dependents          0
estimated_salary        0
calls_made              0
sms_sent                0
data_used               0
churn                   0
dtype: int64
```

[30]:
```
df.dropna(inplace=True)
```

[31]:
```
df.duplicated().sum()
```

[31]:
```
0
```

[32]:
```
df.drop_duplicates(inplace=True)
```

[33]:
```
df.columns
```

[33]:
```
Index(['customer_id', 'telecom_partner', 'gender', 'age', 'state', 'city',
       'pincode', 'date_of_registration', 'num_dependents', 'estimated_salary',
       'calls_made', 'sms_sent', 'data_used', 'churn'],
      dtype='object')
```

[34]:
```
df.
  ↪drop(['customer_id',"state","city","telecom_partner","date_of_registration"],inplace=True,a
```

[35]:
```
df.head()
```

[35]:

	gender	age	pincode	num_dependents	estimated_salary	calls_made	sms_sent	data_used	churn
0	F	25	755597	4	124962	44	45	-361	0
1	F	55	125926	2	130556	62	39	5973	0
2	F	57	423976	0	148828	49	24	193	1
3	M	46	522841	1	38722	80	25	9377	1
4	F	26	740247	2	55098	78	15	1393	0

[36]:
```
le=LabelEncoder()
df["gender"]=le.fit_transform(df["gender"])
```

[37]:
```
avg_salary=df.groupby("age")["estimated_salary"].mean()
avg_salary.head()
```

[37]:
```
age
18    84809.556444
19    84915.670839
20    86253.213203
21    85502.052125
22    85740.667759
Name: estimated_salary, dtype: float64
```

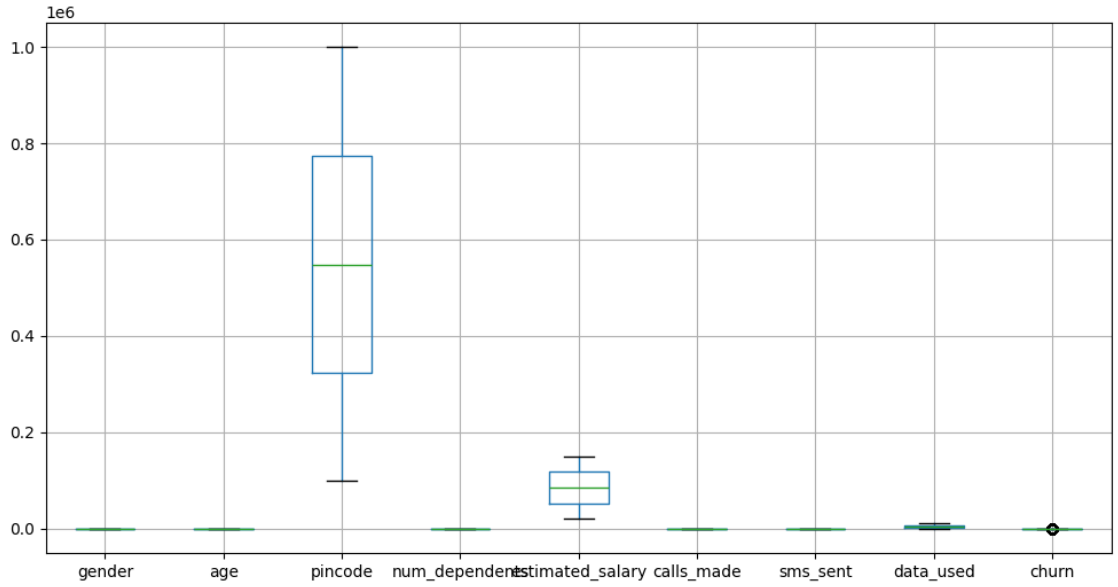
[38]:
```
df.head()
```

[38]:

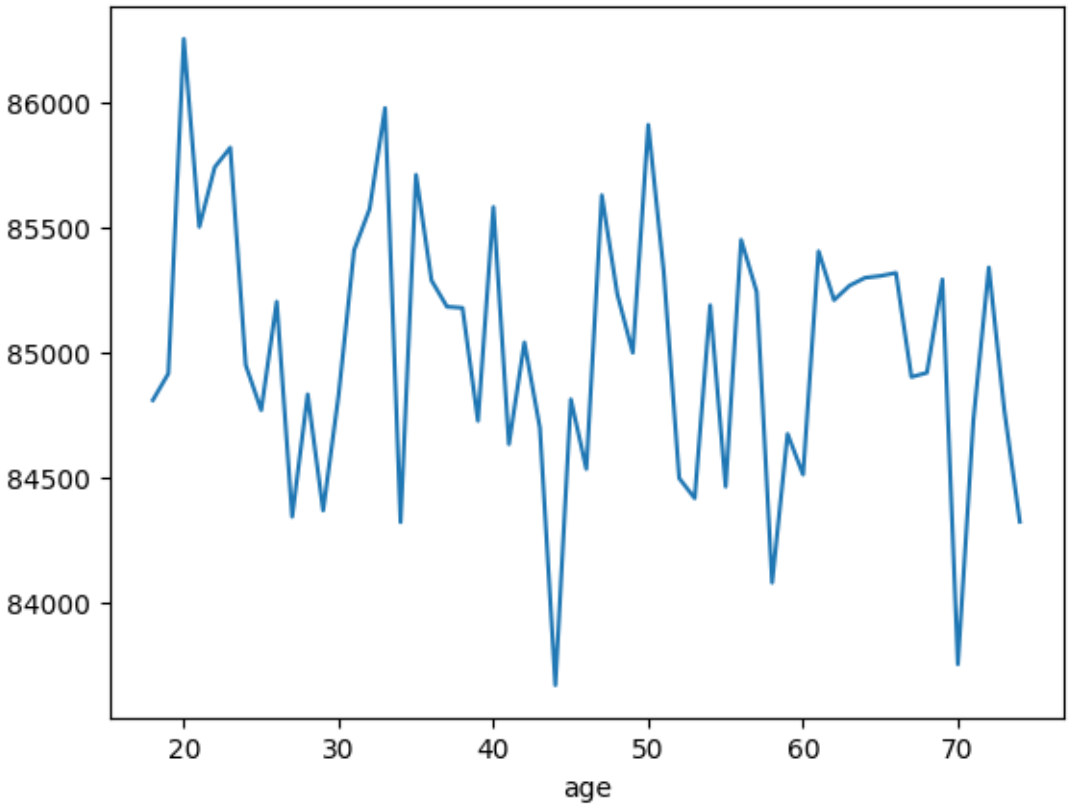
	gender	age	pincode	num_dependents	estimated_salary	calls_made	sms_sent	data_used	churn
0	0	25	755597	4	124962	44	45	-361	0
1	0	55	125926	2	130556	62	39	5973	0
2	0	57	423976	0	148828	49	24	193	1
3	1	46	522841	1	38722	80	25	9377	1
4	0	26	740247	2	55098	78			

```
4          15         1393       0
```

[39]:
```
plt.figure(figsize=(12,6))
df.boxplot()
plt.show()
```



[40]:
```
avg_salary.plot(kind="line")
plt.show()
```



[41]:
```
x=df.drop("churn",axis=1)
y=df["churn"]
```

[42]:
```
df.dtypes
```

[42]:
```
gender              int32
age                 int64
pincode             int64
num_dependents      int64
estimated_salary    int64
calls_made          int64
sms_sent            int64
data_used           int64
churn               int64
dtype: object
```

[43]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
↪2,random_state=42)
```

[44]:
```
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
```

```
x_test=sc.transform(x_test)
```

[45]:
```
df.to_csv("cleaned_telecom_churn.csv",index=False)
```

[ ]:
```

```