NAME – PRIYANSHI ASATI

SAP ID – 500108525

ROLL NO – R2142220982

BATCH- 4 (AIML-NON HONS.)

PREDICTIVE ANALYTICS LAB- 5

Customer Churn Prediction (regression)

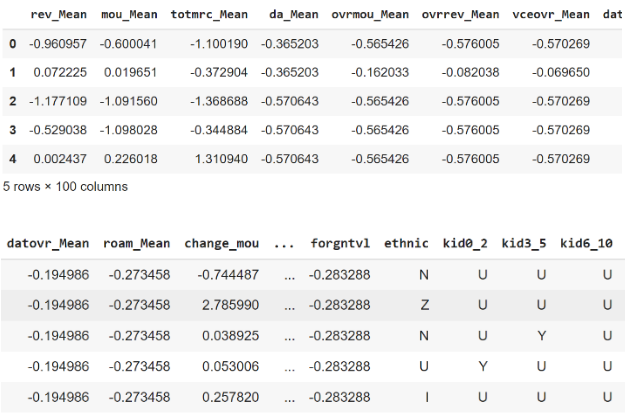
- **Dataset: Telecom customer data (e.g., from Kaggle):**
  - **Data preprocessing (handling missing values, outliers, feature scaling).**

    CODE:

```
import pandas as pd

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import StandardScaler

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

data = pd.read_csv('/content/Telecom_customer churn.csv.zip')

num_cols = data.select_dtypes(include=['float64', 'int64']).columns

imputer_num = SimpleImputer(strategy='mean')

data[num_cols] = imputer_num.fit_transform(data[num_cols])
```

```
cat_cols =
data.select_dtypes(include=['object']).columns

imputer_cat =
SimpleImputer(strategy='most_frequent')

data[cat_cols] =
imputer_cat.fit_transform(data[cat_cols])

z_scores = np.abs((data[num_cols] -
data[num_cols].mean()) / data[num_cols].std())

threshold = 3

data[num_cols] = np.where(z_scores > threshold,
data[num_cols].mean(), data[num_cols])

scaler = StandardScaler()

data[num_cols] =
scaler.fit_transform(data[num_cols])

data.head()
```

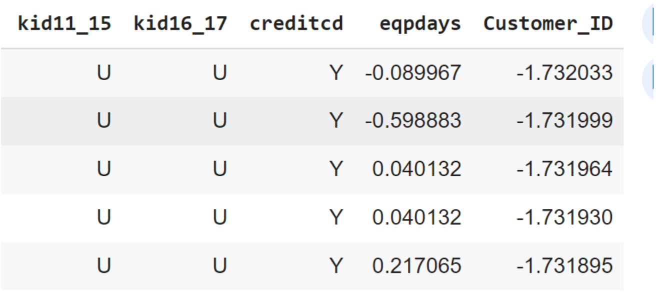
OUTPUT:



	rev_Mean	mou_Mean	totmrc_Mean	da_Mean	ovrmou_Mean	ovrrev_Mean	vceovr_Mean	dat
0	-0.960957	-0.600041	-1.100190	-0.365203	-0.565426	-0.576005	-0.570269	
1	0.072225	0.019651	-0.372904	-0.365203	-0.162033	-0.082038	-0.069650	
2	-1.177109	-1.091560	-1.368688	-0.570643	-0.565426	-0.576005	-0.570269	
3	-0.529038	-1.098028	-0.344884	-0.570643	-0.565426	-0.576005	-0.570269	
4	0.002437	0.226018	1.310940	-0.570643	-0.565426	-0.576005	-0.570269	

5 rows × 100 columns

datovr_Mean	roam_Mean	change_mou	...	forgntvl	ethnic	kid0_2	kid3_5	kid6_10
-0.194986	-0.273458	-0.744487	...	-0.283288	N	U	U	U
-0.194986	-0.273458	2.785990	...	-0.283288	Z	U	U	U
-0.194986	-0.273458	0.038925	...	-0.283288	N	U	Y	U
-0.194986	-0.273458	0.053006	...	-0.283288	U	Y	U	U
-0.194986	-0.273458	0.257820	...	-0.283288	I	U	U	U



kid11_15	kid16_17	creditcd	eqpdays	Customer_ID
U	U	Y	-0.089967	-1.732033
U	U	Y	-0.598883	-1.731999
U	U	Y	0.040132	-1.731964
U	U	Y	0.040132	-1.731930
U	U	Y	0.217065	-1.731895

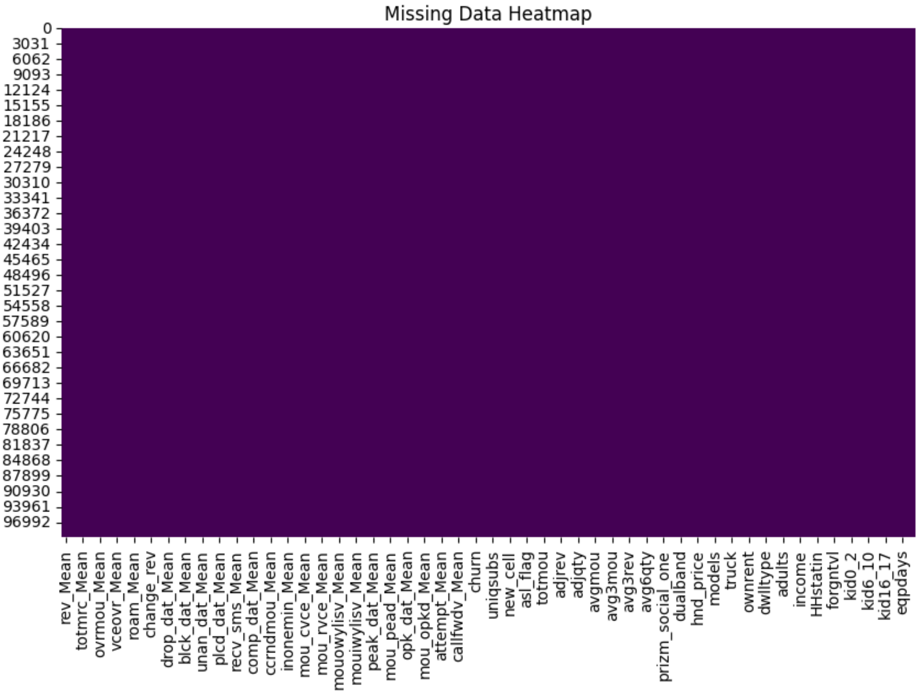
- **Exploratory data analysis (EDA) to identify potential predictors.**

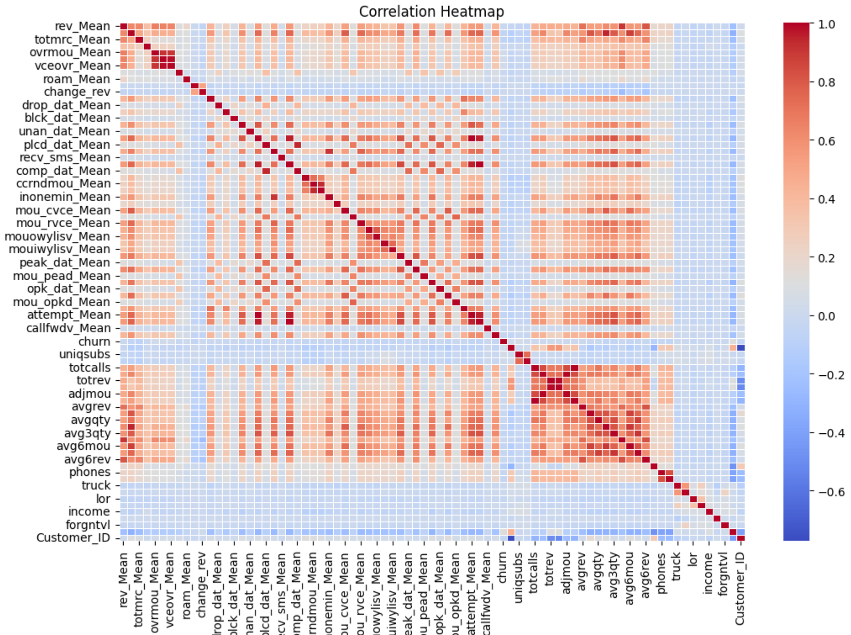
  CODE:

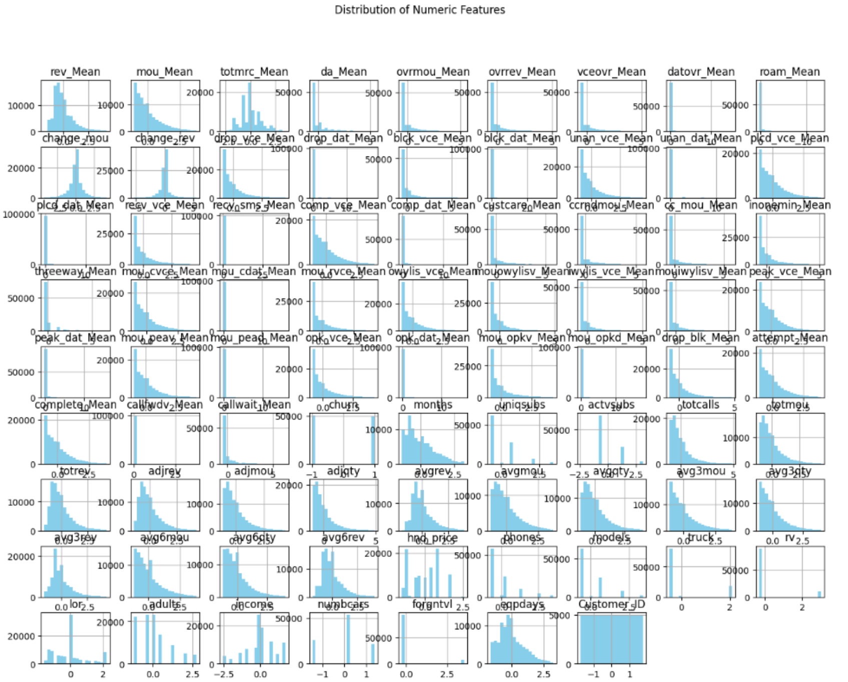
```
plt.figure(figsize=(10, 6))
sns.heatmap(data.isnull(), cbar=False, cmap='viridis')
plt.title('Missing Data Heatmap')
plt.show()
plt.figure(figsize=(12, 8))
correlation_matrix = data[num_cols].corr()
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
num_cols = data.select_dtypes(include=['float64', 'int64']).columns
data[num_cols].hist(bins=20, figsize=(16, 12), color='skyblue')
plt.suptitle('Distribution of Numeric Features')
plt.show()
plt.figure(figsize=(12, 8))
```

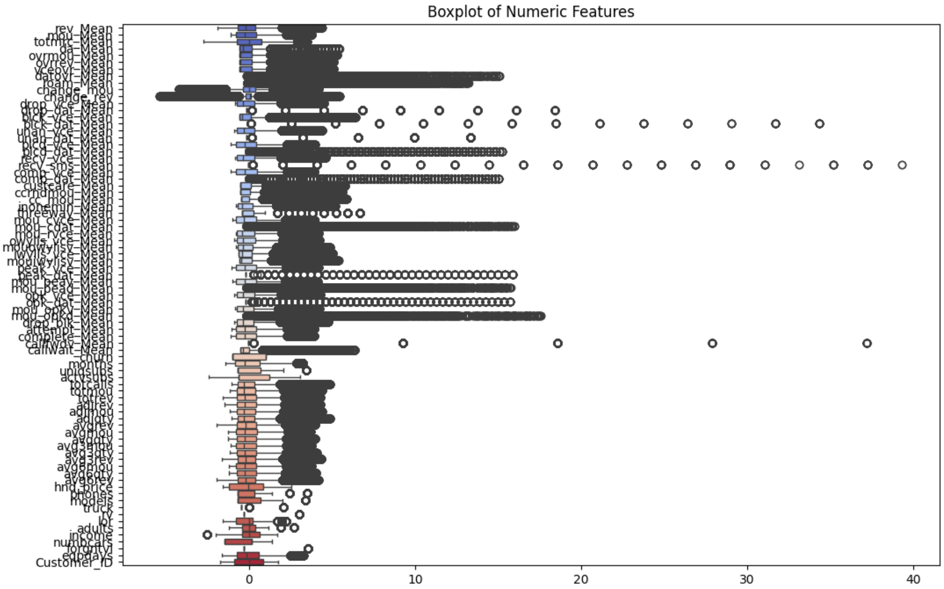
```
sns.boxplot(data=data[num_cols], orient='h', palette='coolwarm')
plt.title('Boxplot of Numeric Features')
plt.show()
top_corr_features = correlation_matrix.abs().unstack().sort_values(ascending=False).drop_duplicates().head(5).index
top_corr_data = data[list(set([i[0] for i in top_corr_features]))]
sns.pairplot(top_corr_data)
plt.suptitle('Pairplot of Top Correlated Features', y=1.02)
plt.show()
cat_cols = data.select_dtypes(include=['object']).columns
for col in cat_cols:
    plt.figure(figsize=(8, 4))
    sns.countplot(x=data[col], palette='coolwarm')
    plt.title(f'Distribution of {col}')
    plt.xticks(rotation=90)
plt.show()
```

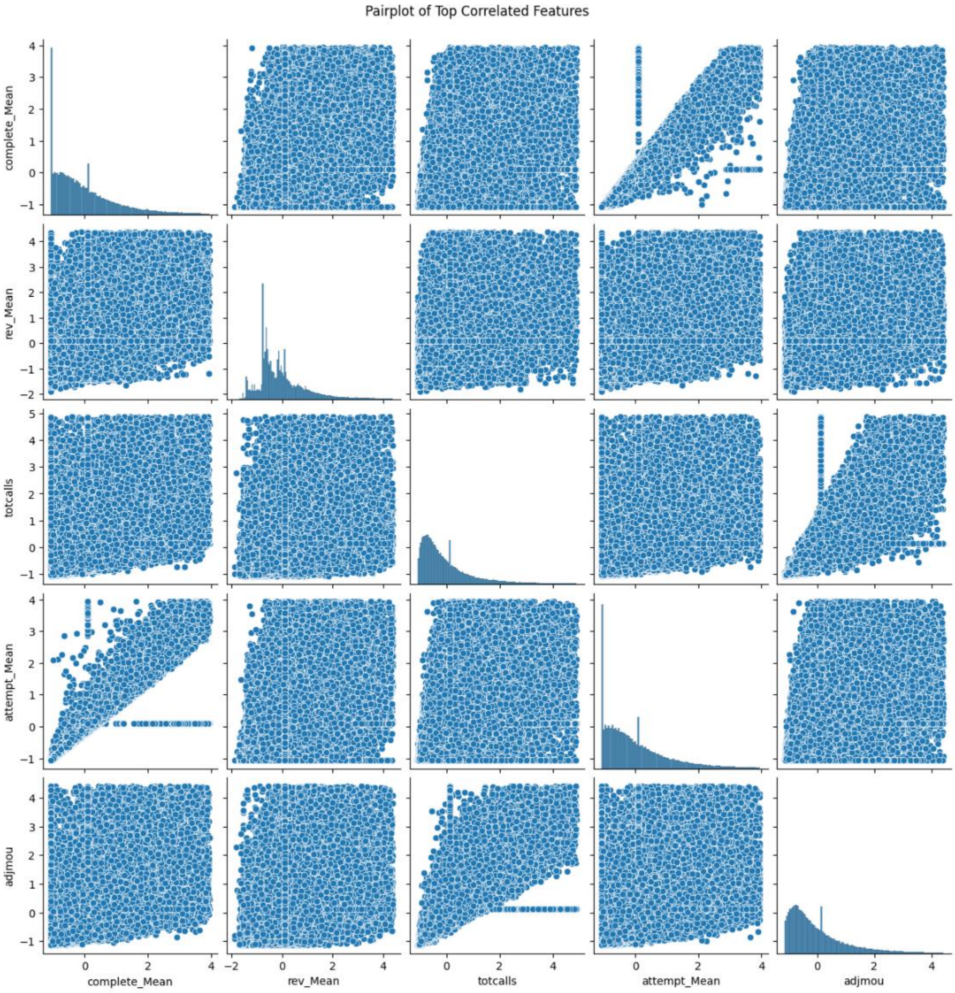
OUTPUT:

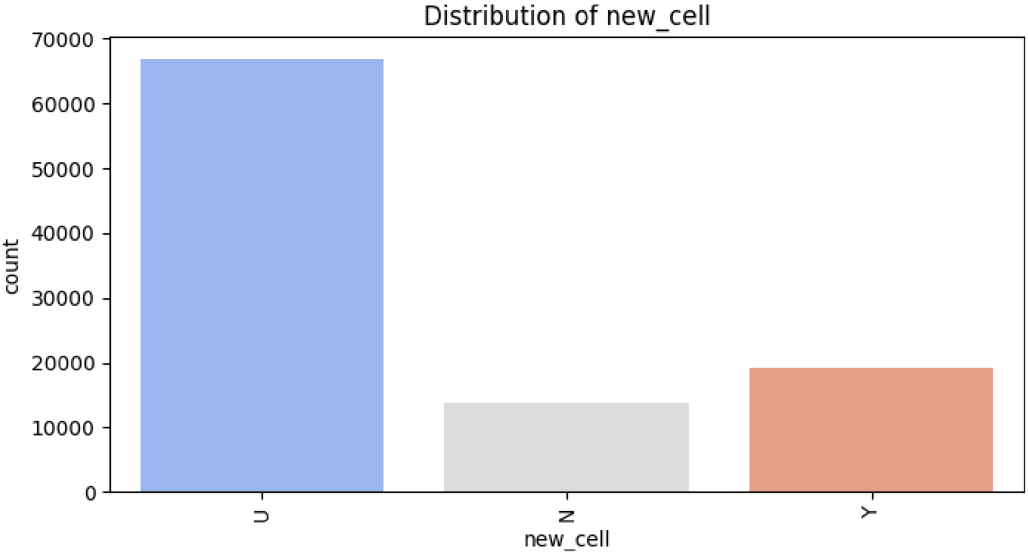

Missing Data Heatmap
0
3031
6062
9093
12124
15155
18186
21217
24248
27279
30310
33341
36372
39403
42434
45465
48496
51527
54558
57589
60620
63651
66682
69713
72744
75775
78806
81837
84868
87899
90930
93961
96992
rev_Mean
totmrc_Mean
ovrmou_Mean
vceovr_Mean
roam_Mean
change_rev
drop_dat_Mean
blck_dat_Mean
unan_dat_Mean
plcd_dat_Mean
recv_sms_Mean
comp_dat_Mean
ccrndmou_Mean
inonemin_Mean
mou_cvce_Mean
mou_rvce_Mean
mouowylisv_Mean
mouiwylisv_Mean
peak_dat_Mean
mou_pead_Mean
opk_dat_Mean
mou_opkd_Mean
attempt_Mean
callfwdv_Mean
churn
uniqsubs
new_cell
asl_flag
totmou
adjrev
adjqty
avgmou
avg3mou
avg3rev
avg6qty
prizm_social_one
dualband
hnd_price
models
truck
ownrent
dwlltype
adults
income
HHstatin
forgntvl
kid0_2
kid6_10
kid16_17
eqpdays

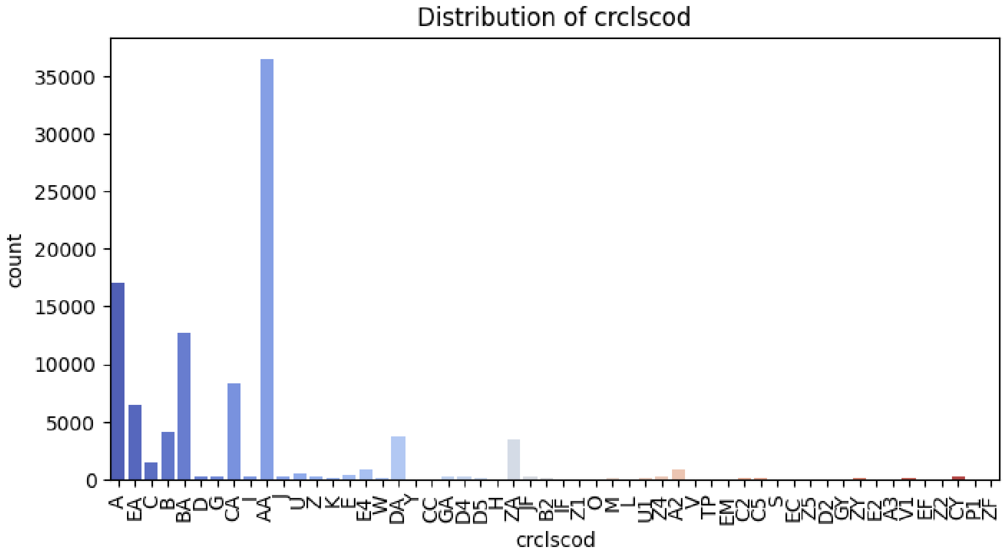

Correlation Heatmap
rev_Mean
totmrc_Mean
ovrmou_Mean
vceovr_Mean
roam_Mean
change_rev
drop_dat_Mean
blck_dat_Mean
unan_dat_Mean
plcd_dat_Mean
recv_sms_Mean
comp_dat_Mean
ccrndmou_Mean
inonemin_Mean
mou_cvce_Mean
mou_rvce_Mean
mouowylisv_Mean
mouiwylisv_Mean
peak_dat_Mean
mou_pead_Mean
opk_dat_Mean
mou_opkd_Mean
attempt_Mean
callfwdv_Mean
churn
uniqsubs
totcalls
totrev
adjmou
avgrev
avgqty
avg3qty
avg6mou
avg6rev
phones
truck
lor
income
forgntvl
Customer_ID
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


Distribution of Numeric Features
rev_Mean
mou_Mean
totmrc_Mean
da_Mean
ovrmou_Mean
ovrrev_Mean
vceovr_Mean
datovr_Mean
roam_Mean
change_mou
change_rev
drop_vce_Mean
drop_dat_Mean
blck_vce_Mean
blck_dat_Mean
unan_vce_Mean
unan_dat_Mean
plcd_vce_Mean
plcd_dat_Mean
recv_vce_Mean
recv_sms_Mean
comp_vce_Mean
comp_dat_Mean
custcare_Mean
ccrndmou_Mean
cc_mou_Mean
inonemin_Mean
threeway_Mean
mou_cvce_Mean
mou_cdat_Mean
mou_rvce_Mean
owylis_vce_Mean
mouowylisv_Mean
iwylis_vce_Mean
mouiwylisv_Mean
peak_vce_Mean
peak_dat_Mean
mou_peav_Mean
mou_pead_Mean
opk_vce_Mean
opk_dat_Mean
mou_opkv_Mean
mou_opkd_Mean
drop_blk_Mean
attempt_Mean
complete_Mean
callfwdv_Mean
callwait_Mean
churn
months
uniqsubs
actvsubs
totcalls
totmou
totrev
adjrev
adjmou
adjqty
avgrev
avgmou
avgqty
avg3mou
avg3qty
avg3rev
avg6mou
avg6qty
avg6rev
hnd_price
phones
models
truck
rv
lor
adults
income
numbcars
forgntvl
eqpdays
Customer_ID

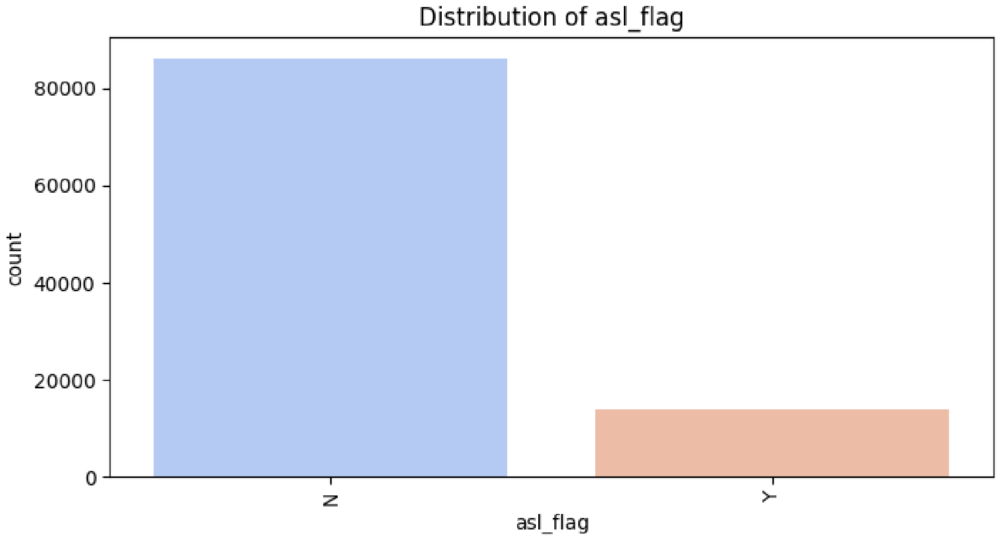

Boxplot of Numeric Features
rev_Mean
mou_Mean
totmrc_Mean
da_Mean
ovrmou_Mean
ovrrev_Mean
vceovr_Mean
datovr_Mean
roam_Mean
change_mou
change_rev
drop_vce_Mean
drop_dat_Mean
blck_vce_Mean
blck_dat_Mean
unan_vce_Mean
unan_dat_Mean
plcd_vce_Mean
plcd_dat_Mean
recv_vce_Mean
recv_sms_Mean
comp_vce_Mean
comp_dat_Mean
custcare_Mean
ccrndmou_Mean
cc_mou_Mean
inonemin_Mean
threeway_Mean
mou_cvce_Mean
mou_cdat_Mean
mou_rvce_Mean
owylis_vce_Mean
mouowylisv_Mean
iwylis_vce_Mean
mouiwylisv_Mean
peak_vce_Mean
peak_dat_Mean
mou_peav_Mean
mou_pead_Mean
opk_vce_Mean
opk_dat_Mean
mou_opkv_Mean
mou_opkd_Mean
drop_blk_Mean
attempt_Mean
complete_Mean
callfwdv_Mean
callwait_Mean
churn
months
uniqsubs
actvsubs
totcalls
totmou
totrev
adjrev
adjmou
adjqty
avgrev
avgmou
avgqty
avg3mou
avg3qty
avg3rev
avg6mou
avg6qty
avg6rev
hnd_price
phones
models
truck
rv
lor
adults
income
numbcars
forgntvl
eqpdays
Customer_ID
0
10
20
30
40


Pairplot of Top Correlated Features
complete_Mean
rev_Mean
totcalls
attempt_Mean
adjmou
complete_Mean
rev_Mean
totcalls
attempt_Mean
adjmou

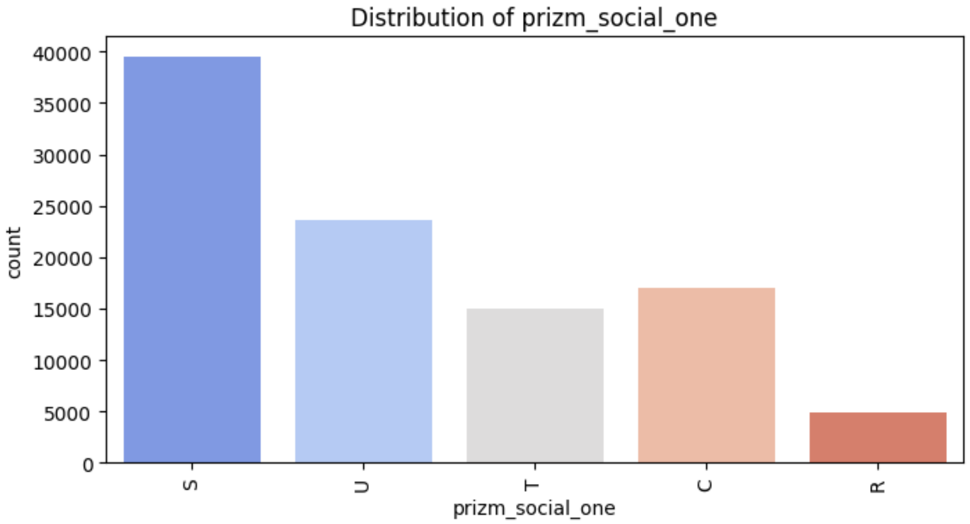

Distribution of new_cell
count
70000
60000
50000
40000
30000
20000
10000
0
U
N
Y
new_cell



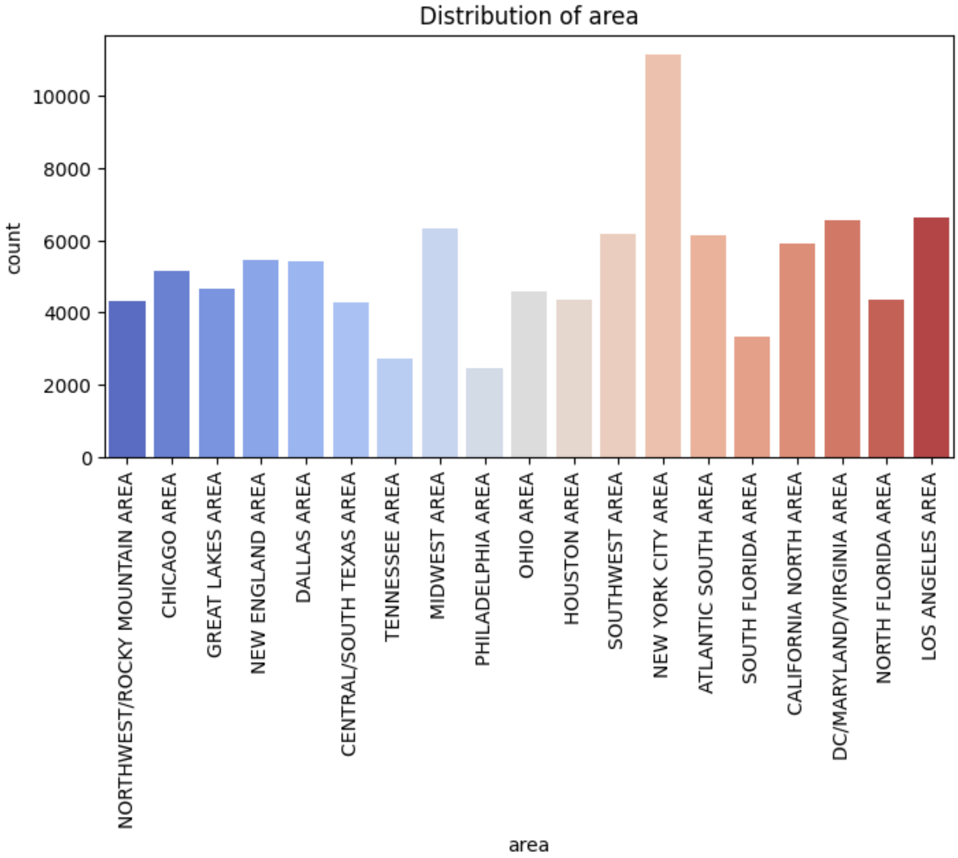
Distribution of crclscod

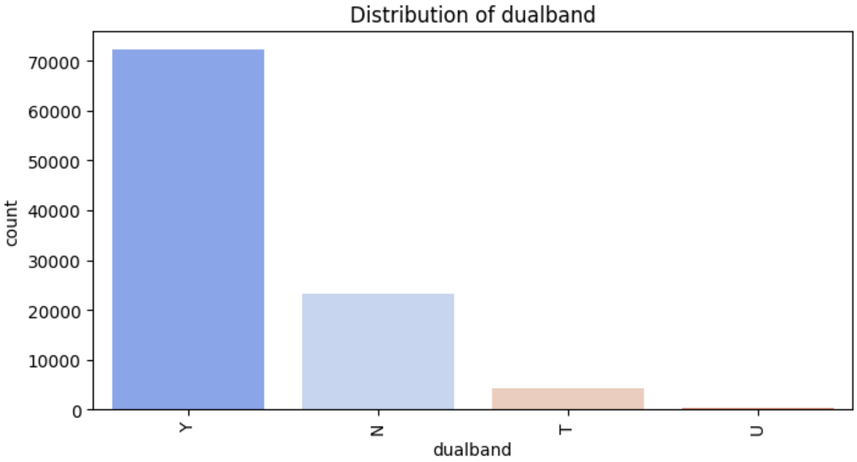


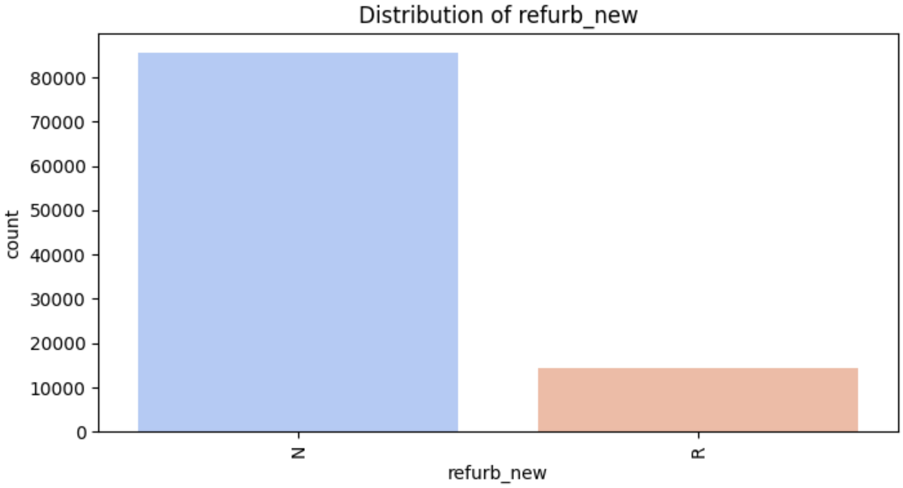
Distribution of asl_flag

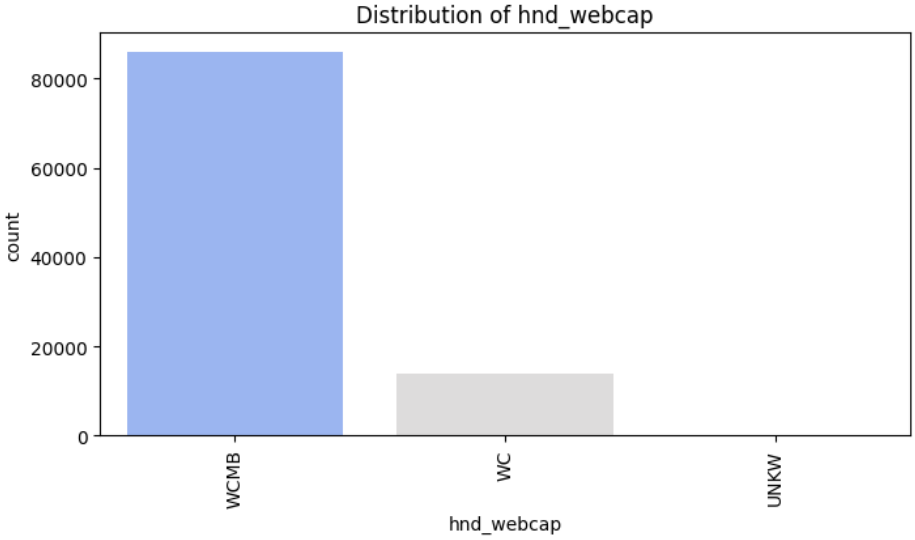


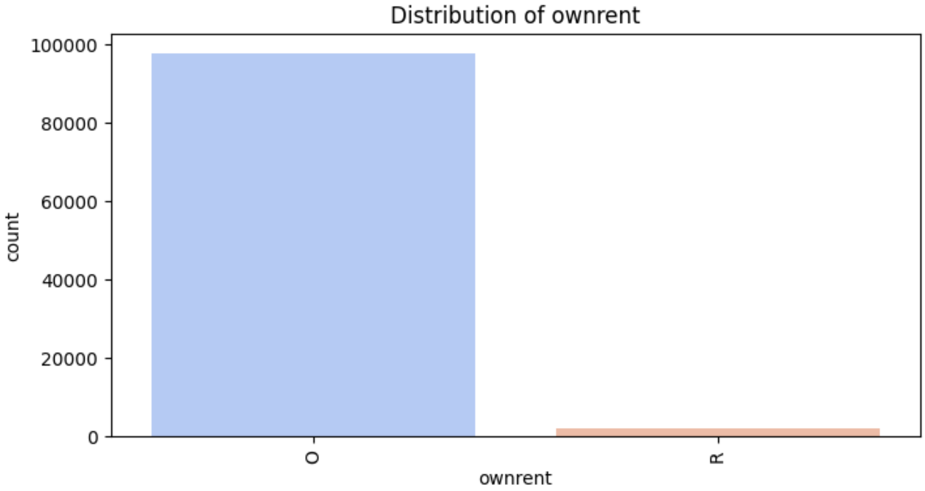
Distribution of prizm_social_one

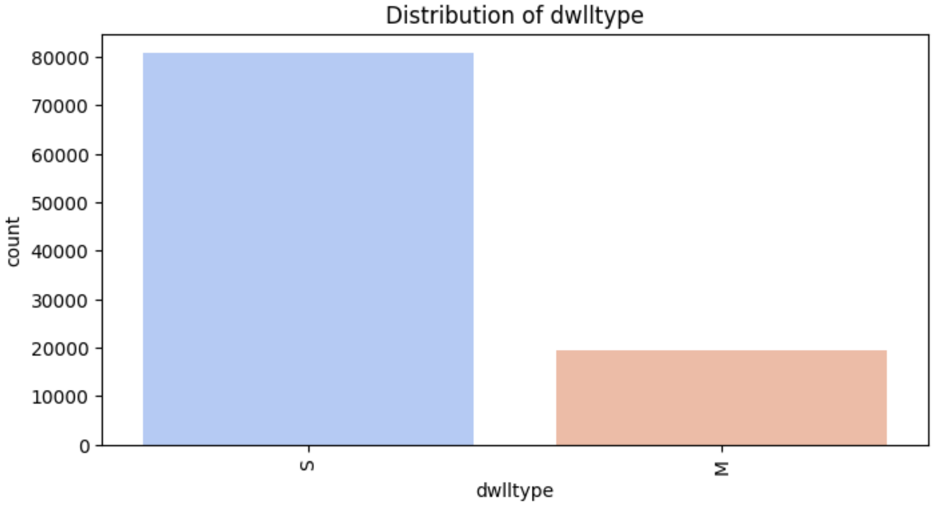

Distribution of area
count
10000
8000
6000
4000
2000
0
NORTHWEST/ROCKY MOUNTAIN AREA
CHICAGO AREA
GREAT LAKES AREA
NEW ENGLAND AREA
DALLAS AREA
CENTRAL/SOUTH TEXAS AREA
TENNESSEE AREA
MIDWEST AREA
PHILADELPHIA AREA
OHIO AREA
HOUSTON AREA
SOUTHWEST AREA
NEW YORK CITY AREA
ATLANTIC SOUTH AREA
SOUTH FLORIDA AREA
CALIFORNIA NORTH AREA
DC/MARYLAND/VIRGINIA AREA
NORTH FLORIDA AREA
LOS ANGELES AREA
area


Distribution of dualband
count
70000
60000
50000
40000
30000
20000
10000
0
Y
N
T
U
dualband

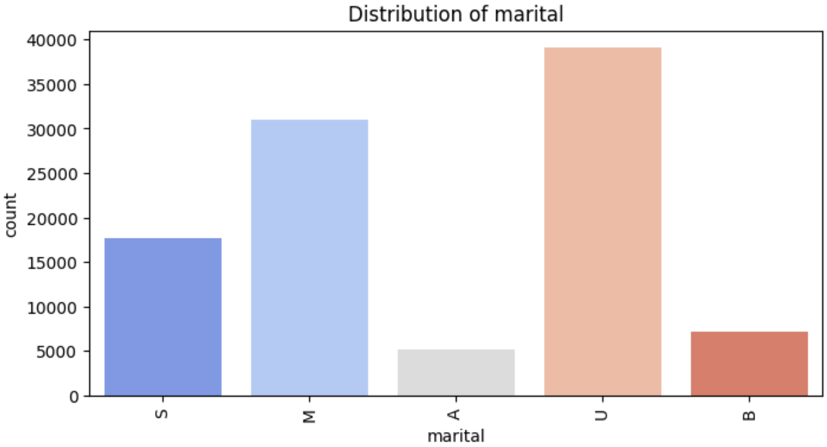

Distribution of refurb_new
count
80000
70000
60000
50000
40000
30000
20000
10000
0
N
R
refurb_new


Distribution of hnd_webcap
count
80000
60000
40000
20000
0
WCMB
WC
UNKW
hnd_webcap

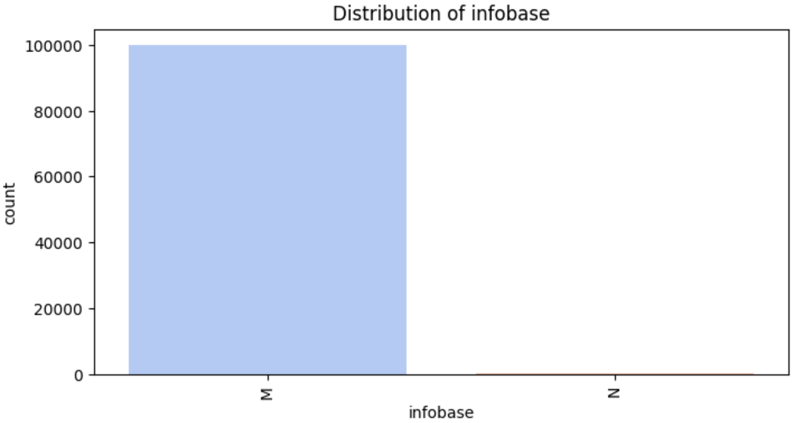

Distribution of ownrent
count
100000
80000
60000
40000
20000
0
O
R
ownrent



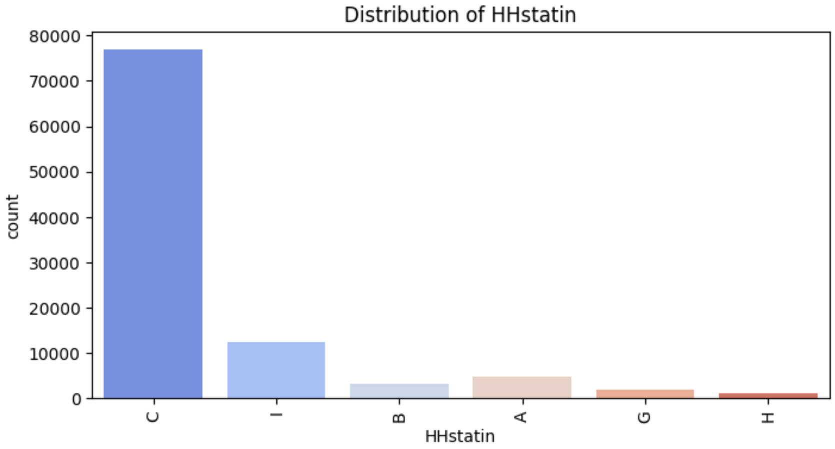
Distribution of dwlltype
count
80000
70000
60000
50000
40000
30000
20000
10000
0
S
M
dwlltype

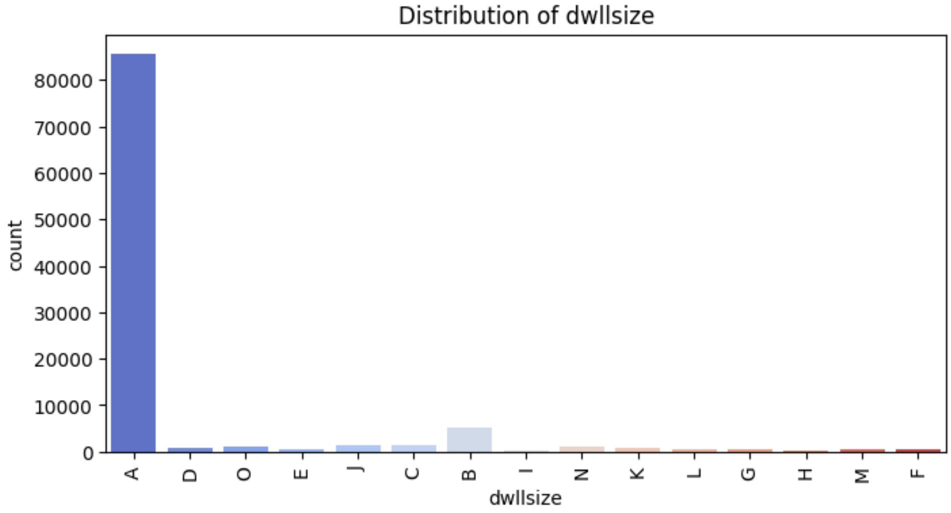


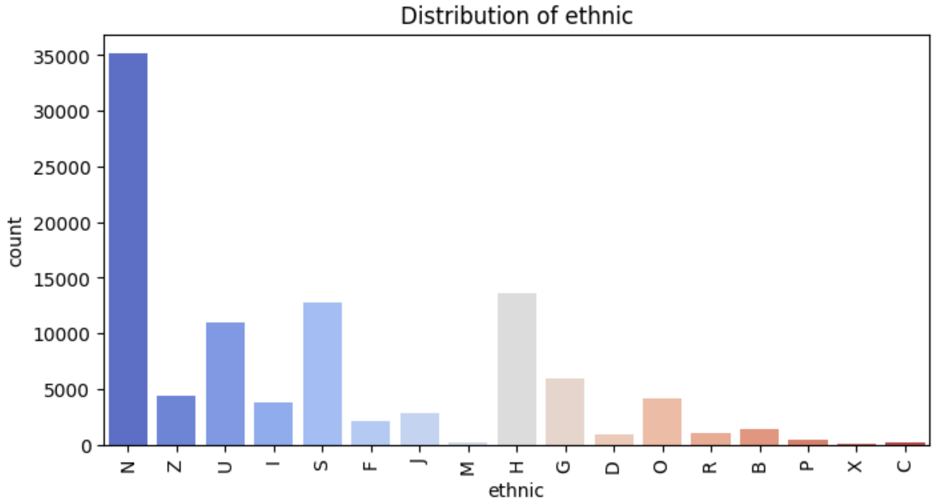
Distribution of marital
count
40000
35000
30000
25000
20000
15000
10000
5000
0
S
M
A
U
B
marital

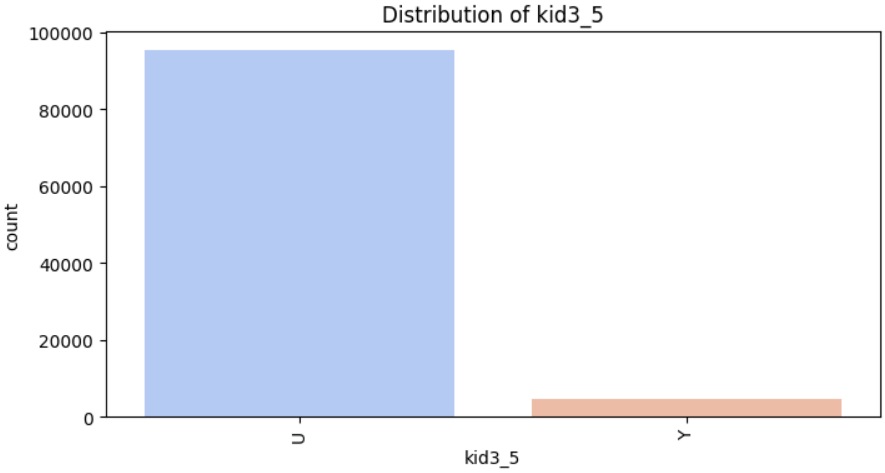


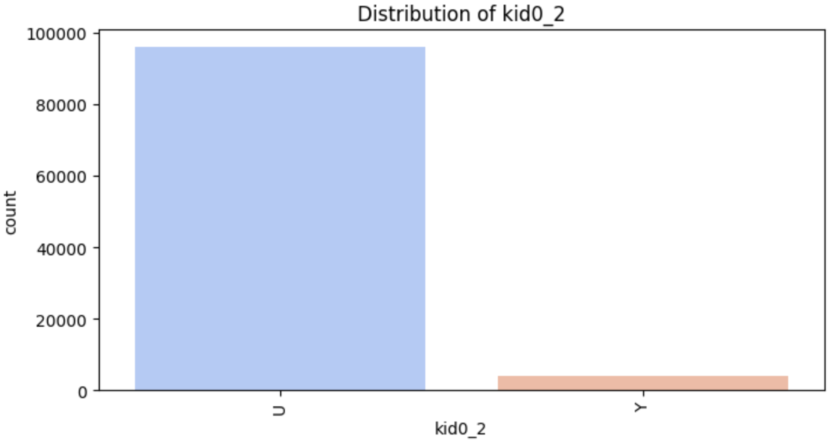
Distribution of infobase
count
100000
80000
60000
40000
20000
0
M
N
infobase

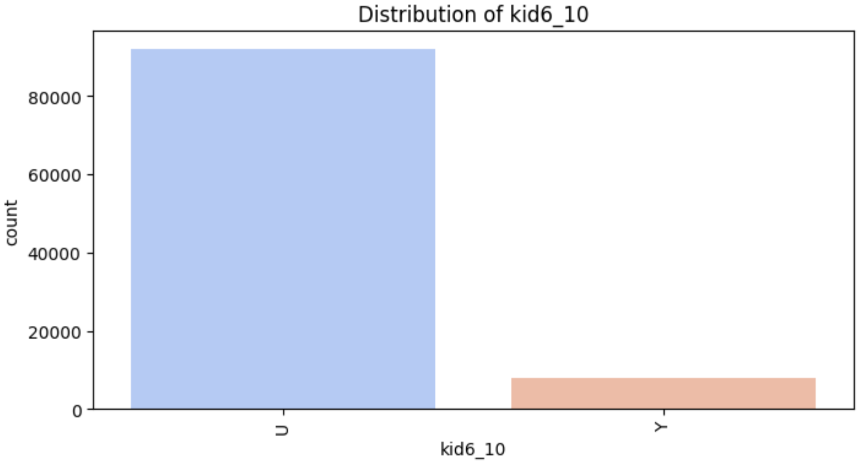

Distribution of HHstatin
count
0
10000
20000
30000
40000
50000
60000
70000
80000
C
I
B
A
G
H
HHstatin

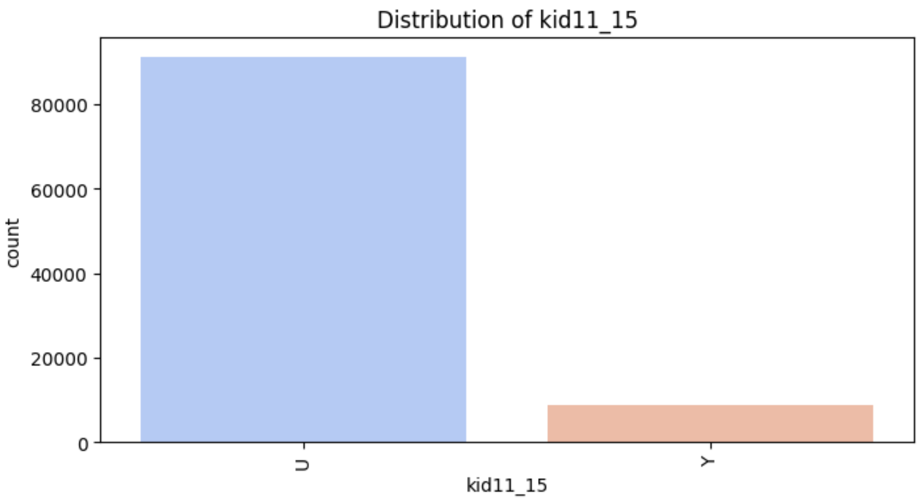

Distribution of dwllsize
count
0
10000
20000
30000
40000
50000
60000
70000
80000
A
D
O
E
J
C
B
I
N
K
L
G
H
M
F
dwllsize

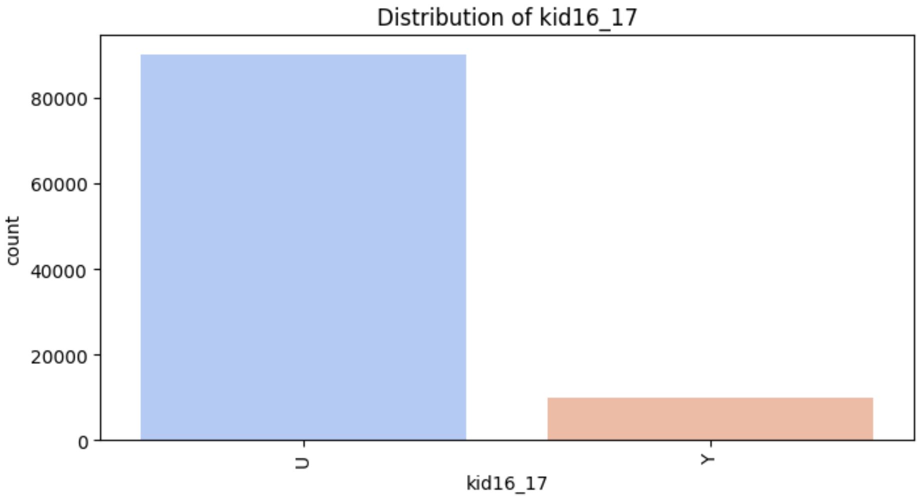

Distribution of ethnic
count
0
5000
10000
15000
20000
25000
30000
35000
N
Z
U
I
S
F
J
M
H
G
D
O
R
B
P
X
C
ethnic

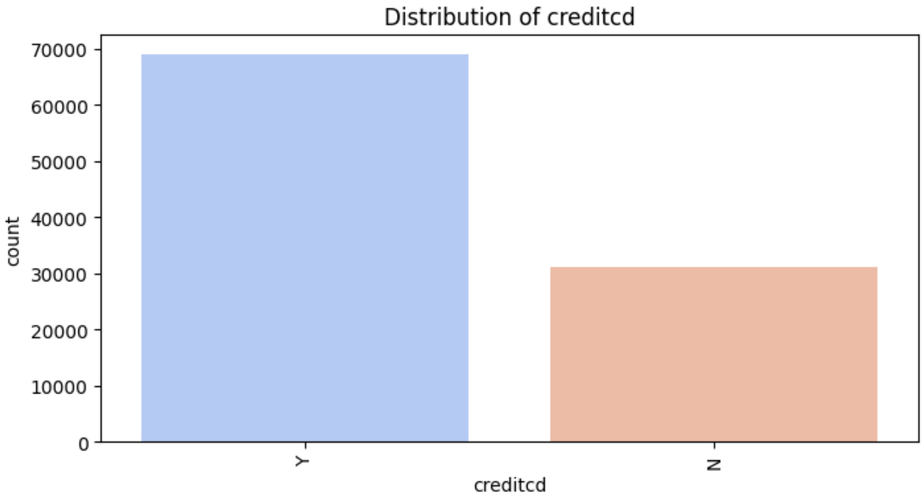

Distribution of kid3_5
100000
80000
60000
40000
20000
0
count
U
Y
kid3_5


Distribution of kid0_2
100000
80000
60000
40000
20000
0
count
U
Y
kid0_2


Distribution of kid6_10
80000
60000
40000
20000
0
count
U
Y
kid6_10





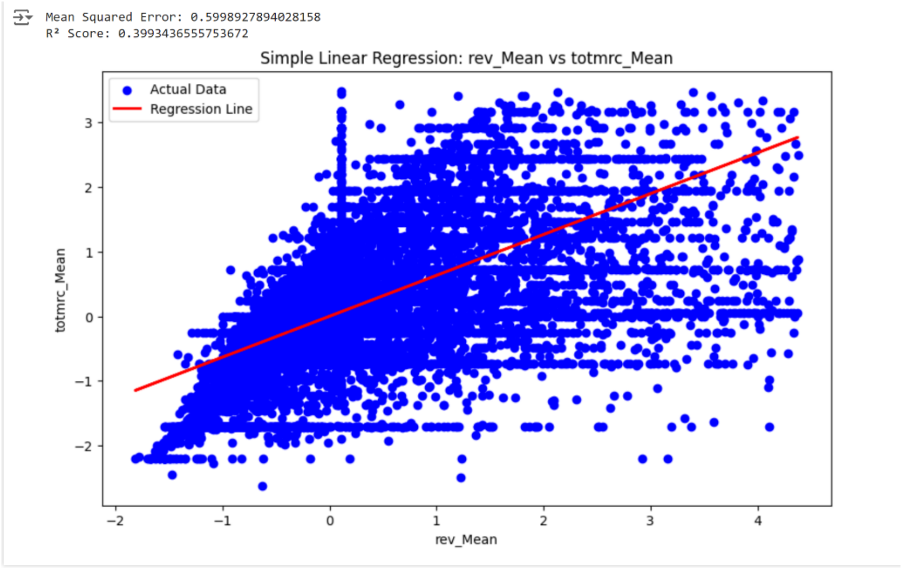


- **Building a simple linear regression model.**

CODE:

```
X = data[['rev_Mean']]
y = data['totmrc_Mean']
X = X.dropna()
y = y[X.index]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
print(f'R² Score: {r2}')
plt.figure(figsize=(10, 6))
plt.scatter(X_test, y_test, color='blue', label='Actual Data')
plt.plot(X_test, y_pred, color='red', linewidth=2, label='Regression Line')
plt.title('Simple Linear Regression: rev_Mean vs totmrc_Mean')
plt.xlabel('rev_Mean')
plt.ylabel('totmrc_Mean')
plt.legend()
plt.show()
```

OUTPUT:



```
Mean Squared Error: 0.5998927894028158
R² Score: 0.3993436555753672
```

- **Evaluating model performance using accuracy, precision, recall, F1-score.**

CODE:

X = data[['rev_Mean'

y = data['churn']

X = X.dropna()

y = y[X.index]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LogisticRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

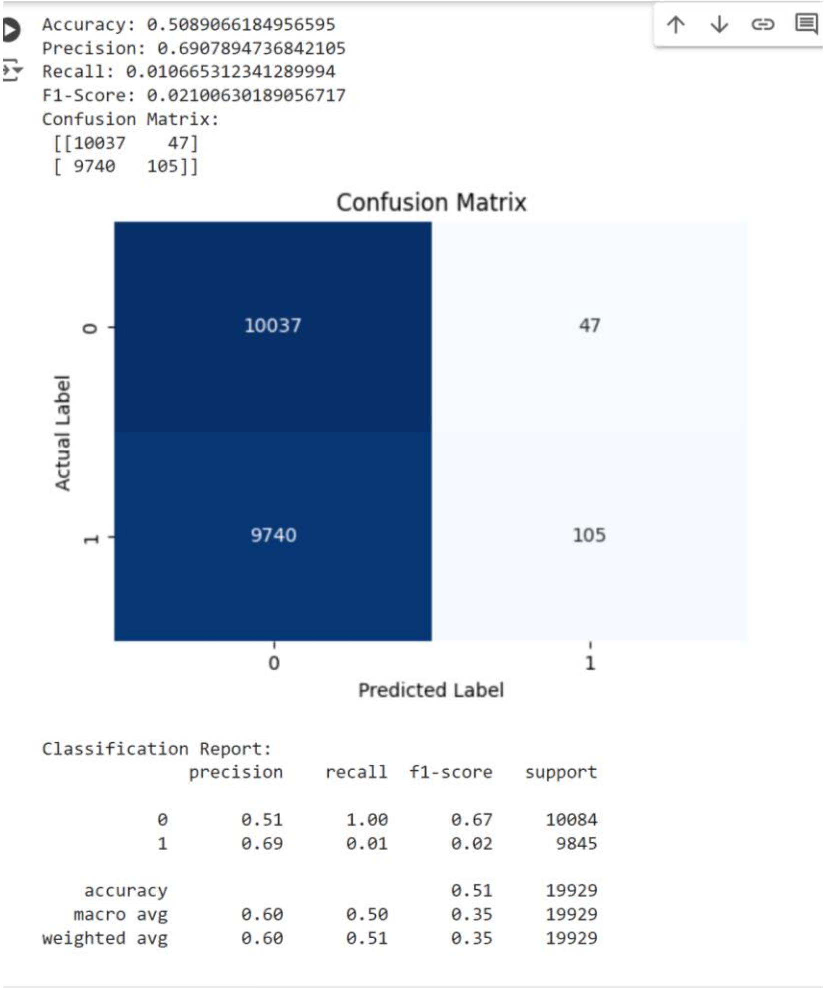
accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy}')

precision = precision_score(y_test, y_pred)

```
print(f'Precision: {precision}')
recall = recall_score(y_test, y_pred)
print(f'Recall: {recall}')
f1 = f1_score(y_test, y_pred)
print(f'F1-Score: {f1}')
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix:\n {conf_matrix}')
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.title('Confusion Matrix')
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.show()
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

OUTPUT:



```
Accuracy: 0.5089066184956595
Precision: 0.6907894736842105
Recall: 0.010665312341289994
F1-Score: 0.02100630189056717
Confusion Matrix:
 [[10037    47]
 [ 9740   105]]
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.51      1.00      0.67     10084
           1       0.69      0.01      0.02      9845

    accuracy                           0.51     19929
   macro avg       0.60      0.50      0.35     19929
weighted avg       0.60      0.51      0.35     19929
```