OBJECTIVE

Our objective is to predict CUSTOMER LIFETIME VALUE for an Insurance Company. CLV is total worth to a Business given by customer over a period of their relationship.

ABC Insurance company has a problem in retaining its customer and wants to advertise promotional offer to its loyal customers . For this purpose , company should use CLV.

CODE FOR THE DATASET

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
InsuranceData=pd.read_csv("insurance.csv")
```

In [3]:
```
InsuranceData
```

Out[3]:

	Customer	State	Customer Lifetime Value	Response	Coverage	Education	Effective To Date	Employme
0	BU79786	Washington	2763.519279	No	Basic	Bachelor	2/24/11	
1	QZ44356	Arizona	6979.535903	No	Extended	Bachelor	1/31/11	Un
2	AI49188	Nevada	12887.431650	No	Premium	Bachelor	2/19/11	
3	WW63253	California	7645.861827	No	Basic	Bachelor	1/20/11	Un
4	HB64268	Washington	2813.692575	No	Basic	Bachelor	02-03-2011	
...	...	...	...	...	...	...	...	
9129	LA72316	California	23405.987980	No	Basic	Bachelor	02-10-2011	
9130	PK87824	California	3096.511217	Yes	Extended	College	02-12-2011	
9131	TD14365	California	8163.890428	No	Extended	Bachelor	02-06-2011	Un
9132	UP19263	California	7524.442436	No	Extended	College	02-03-2011	
9133	Y167826	California	2611.836866	No	Extended	College	2/14/11	Un

9134 rows × 24 columns

Columns/Features are:

1. Customer ID
2. State : place where customer lives
3. Customer lifetime value till now in dollar
4. Response: whether or not customer going to get another policy
5. Coerage : Basic , extended, premium whatever company has
6. Education
7. Effective to date of insurance policy: date on which policy went i nto effect,i.e., date of renewing the policy
8. Employment status
9. Gender
10. Income
11. Location code
12. Marital status
13. Monthly premium auto
14. Months since last claim
15. Months since policy Inception : month since the policy was first purchased
16. No. of open complaints
17. No. of policies
18. Policy type
19. Policy
20. Renew offer type
21. Sales channel
22. Total claim amount: in dollar
23. Vehicle class
24. Vehicle size

In [4]:
```
# Checking top 6 observations of the dataset
print(InsuranceData.head(6))

# Checking bottom 6 observations of the dataset
print(InsuranceData.tail(6))
```

```
  Customer       State  Customer Lifetime Value Response  Coverage Education
\
0  BU79786  Washington              2763.519279       No     Basic  Bachelor
1  QZ44356     Arizona              6979.535903       No  Extended  Bachelor
2  AI49188      Nevada             12887.431650       No   Premium  Bachelor
3  WW63253  California              7645.861827       No     Basic  Bachelor
4  HB64268  Washington              2813.692575       No     Basic  Bachelor
5  OC83172      Oregon              8256.297800      Yes     Basic  Bachelor

  Effective To Date EmploymentStatus Gender  Income  ...  \
0           2/24/11         Employed      F   56274  ...
1           1/31/11       Unemployed      F       0  ...
2           2/19/11         Employed      F   48767  ...
3           1/20/11       Unemployed      M       0  ...
4        02-03-2011         Employed      M   43836  ...
5           1/25/11         Employed      F   62902  ...

  Months Since Policy Inception Number of Open Complaints  Number of Policies
\
0                             5                         0                   1
1                            42                         0                   8
2                            38                         0                   2
3                            65                         0                   7
4                            44                         0                   1
5                            94                         0                   2

      Policy Type        Policy  Renew Offer Type  Sales Channel  \
0  Corporate Auto  Corporate L3            Offer1          Agent
1   Personal Auto   Personal L3            Offer3          Agent
2   Personal Auto   Personal L3            Offer1          Agent
3  Corporate Auto  Corporate L2            Offer1    Call Center
4   Personal Auto   Personal L1            Offer1          Agent
5   Personal Auto   Personal L3            Offer2            Web

  Total Claim Amount  Vehicle Class Vehicle Size
0         384.811147   Two-Door Car      Medsize
1        1131.464935  Four-Door Car      Medsize
2         566.472247   Two-Door Car      Medsize
3         529.881344            SUV      Medsize
4         138.130879  Four-Door Car      Medsize
5         159.383042   Two-Door Car      Medsize

[6 rows x 24 columns]
     Customer       State  Customer Lifetime Value Response  Coverage  \
9128  YM19146  California              4100.398533       No   Premium
9129  LA72316  California             23405.987980       No     Basic
9130  PK87824  California              3096.511217      Yes  Extended
9131  TD14365  California              8163.890428       No  Extended
9132  UP19263  California              7524.442436       No  Extended
9133  Y167826  California              2611.836866       No  Extended

     Education Effective To Date EmploymentStatus Gender  Income  ...  \
9128   College        01-06-2011         Employed      F   47761  ...
9129  Bachelor        02-10-2011         Employed      M   71941  ...
9130   College        02-12-2011         Employed      F   21604  ...
9131  Bachelor        02-06-2011       Unemployed      M       0  ...
9132   College        02-03-2011         Employed      M   21941  ...
```

```
9133   College           2/14/11       Unemployed      M       0  ...

     Months Since Policy Inception Number of Open Complaints  \
9128                             58                         0
9129                             89                         0
9130                             28                         0
9131                             37                         3
9132                              3                         0
9133                             90                         0

      Number of Policies     Policy Type        Policy  Renew Offer Type  \
9128                   1   Personal Auto   Personal L2            Offer1
9129                   2   Personal Auto   Personal L1            Offer2
9130                   1  Corporate Auto  Corporate L3            Offer1
9131                   2  Corporate Auto  Corporate L2            Offer1
9132                   3   Personal Auto   Personal L2            Offer3
9133                   1  Corporate Auto  Corporate L3            Offer4

      Sales Channel Total Claim Amount  Vehicle Class Vehicle Size
9128         Branch         541.282007  Four-Door Car        Large
9129            Web         198.234764  Four-Door Car      Medsize
9130         Branch         379.200000  Four-Door Car      Medsize
9131         Branch         790.784983  Four-Door Car      Medsize
9132         Branch         691.200000  Four-Door Car        Large
9133    Call Center         369.600000   Two-Door Car      Medsize

[6 rows x 24 columns]
```

In [5]:
```
InsuranceData = InsuranceData.drop(columns=["Customer"])
```

In [6]:
```
InsuranceData.columns
```

Out[6]:
```
Index(['State', 'Customer Lifetime Value', 'Response', 'Coverage', 'Educatio
n',
       'Effective To Date', 'EmploymentStatus', 'Gender', 'Income',
       'Location Code', 'Marital Status', 'Monthly Premium Auto',
       'Months Since Last Claim', 'Months Since Policy Inception',
       'Number of Open Complaints', 'Number of Policies', 'Policy Type',
       'Policy', 'Renew Offer Type', 'Sales Channel', 'Total Claim Amount',
       'Vehicle Class', 'Vehicle Size'],
      dtype='object')
```

In [7]:
```
# Dimensions of the dataset
print(InsuranceData.shape)

# Structure of the dataset
print(InsuranceData.info())
```

```
(9134, 23)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9134 entries, 0 to 9133
Data columns (total 23 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   State                          9134 non-null   object
 1   Customer Lifetime Value        9134 non-null   float64
 2   Response                       9134 non-null   object
 3   Coverage                       9134 non-null   object
 4   Education                      9134 non-null   object
 5   Effective To Date              9134 non-null   object
 6   EmploymentStatus               9134 non-null   object
 7   Gender                         9134 non-null   object
 8   Income                         9134 non-null   int64
 9   Location Code                  9134 non-null   object
 10  Marital Status                 9134 non-null   object
 11  Monthly Premium Auto           9134 non-null   int64
 12  Months Since Last Claim        9134 non-null   int64
 13  Months Since Policy Inception  9134 non-null   int64
 14  Number of Open Complaints      9134 non-null   int64
 15  Number of Policies             9134 non-null   int64
 16  Policy Type                    9134 non-null   object
 17  Policy                         9134 non-null   object
 18  Renew Offer Type               9134 non-null   object
 19  Sales Channel                  9134 non-null   object
 20  Total Claim Amount             9134 non-null   float64
 21  Vehicle Class                  9134 non-null   object
 22  Vehicle Size                   9134 non-null   object
dtypes: float64(2), int64(6), object(15)
memory usage: 1.6+ MB
None
```

The provided dataset has lots of details :

1. There are 9134 Observations of 24 Variable
2. There are mix of categorical and continous DataType.
3. Dependent Variable is Customer Life Time Value as we have to predict the CLV.
4. Independent Variables are: Customer, StateCustomerLifetimeValue, Response, Coverage, Education, EffectiveToDate, EmploymentStatus, Gender, Income, LocationCode, MaritalStatus, MonthlyPremiumAuto, MonthsSinceLastClaim, MonthsSincePolicyInception, NumberofOpenComplaints, NumberofPoliciesPolicyType, Policy, RenewOfferType, SalesChannel, TotalClaimAmountVehicleClass, VehicleSize
5. Continues Independed Variables are : CustomerLifetimeValue, Income,MonthlyPremiumAuto, MonthsSinceLastClaim, MonthsSincePolicyInception, NumberofOpenComplaints, NumberofPolicies, TotalClaimAmount
6. There are no null values, so no further action required to replace missing or null values.

7. “Customer” column is serial number so it is insignificat for analysis and removed from the dataset.

In [8]:
```
# Checking null values in each column
na_counts = InsuranceData.isnull().sum()

# Creating a data frame with the null value counts
na_counts = pd.DataFrame(na_counts, columns=["na_counts"])

print(na_counts)
```

```
                               na_counts
State                                  0
Customer Lifetime Value                0
Response                               0
Coverage                               0
Education                              0
Effective To Date                      0
EmploymentStatus                       0
Gender                                 0
Income                                 0
Location Code                          0
Marital Status                         0
Monthly Premium Auto                   0
Months Since Last Claim                0
Months Since Policy Inception          0
Number of Open Complaints              0
Number of Policies                     0
Policy Type                            0
Policy                                 0
Renew Offer Type                       0
Sales Channel                          0
Total Claim Amount                     0
Vehicle Class                          0
Vehicle Size                           0
```

There are no null values, so no further action required to replace missing or null values.

In [9]:
```
# Compute the number of unique values in each column
unique_counts = InsuranceData.apply(lambda x: np.unique(x).shape[0])

# Print the unique value counts
print(unique_counts)
```

```
State                             5
Customer Lifetime Value        8041
Response                          2
Coverage                          3
Education                         5
Effective To Date                59
EmploymentStatus                  5
Gender                            2
Income                         5694
Location Code                     3
Marital Status                    3
Monthly Premium Auto            202
Months Since Last Claim          36
Months Since Policy Inception   100
Number of Open Complaints         6
Number of Policies                9
Policy Type                       3
Policy                            9
Renew Offer Type                  4
Sales Channel                     4
Total Claim Amount             5106
Vehicle Class                     6
Vehicle Size                      3
dtype: int64
```

In [10]:
```
# FINDING THE  NO. UNIQUE VALUES IN EACH COLUMN
l=["State","Response","Coverage","Education","EmploymentStatus","Gender",
   "Location Code","Marital Status","Policy Type","Policy","Renew Offer Type",
   "Sales Channel","Vehicle Class","Vehicle Size"]
for i in l:
    print("Unique values of: ",i )
    print( InsuranceData[i].value_counts())
    print("\n")
```

```
Unique values of:  State
California    3150
Oregon        2601
Arizona       1703
Nevada         882
Washington     798
Name: State, dtype: int64


Unique values of:  Response
No     7826
Yes    1308
Name: Response, dtype: int64


Unique values of:  Coverage
Basic       5568
Extended    2742
Premium      824
```

Hence, we have found that there is no nul or NaN values and also the number of unique values.

Now, we will move forward on EDA.

In this section we perform initial investigations on insurance data so as to discover patterns and to check assumptions with the help of summary statistics and graphical representations.

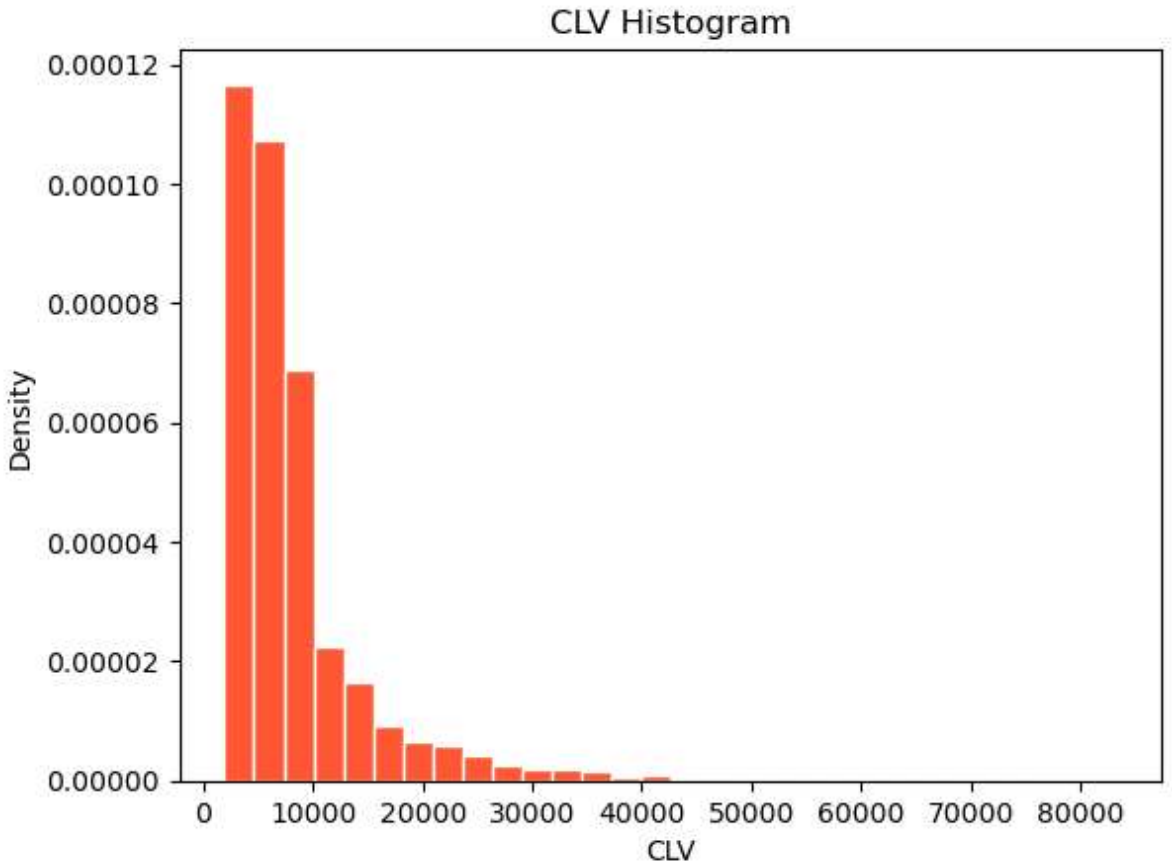
Exploratory Data Analysis

DESCRIPTIVE ANALYSIS of CustomerLifetimeValue

In [11]:
```
# Calculate the range, mean, standard deviation, summary statistics
clv_values = InsuranceData['Customer Lifetime Value']

clv_range = np.ptp(clv_values)
clv_mean = np.mean(clv_values)
clv_sd = np.std(clv_values)
clv_summary = clv_values.describe()

# Calculate variance, skewness, and kurtosis
clv_variance = np.var(clv_values)
clv_skewness = clv_values.skew()
clv_kurtosis = clv_values.kurtosis()

# Print the results
print("Range:", clv_range)
print("Mean:", clv_mean)
print("Standard Deviation:", clv_sd)
print("Summary Statistics:")
print(clv_summary)
print("Variance:", clv_variance)
print("Skewness:", clv_skewness)
print("Kurtosis:", clv_kurtosis)
```

```
Range: 81427.373515
Mean: 8004.940474987081
Standard Deviation: 6870.5914776541185
Summary Statistics:
count     9134.000000
mean      8004.940475
std       6870.967608
min       1898.007675
25%       3994.251794
50%       5780.182197
75%       8962.167041
max      83325.381190
Name: Customer Lifetime Value, dtype: float64
Variance: 47205027.25281341
Skewness: 3.0322802599299847
Kurtosis: 13.8235334254424
```

In [19]:
```
import matplotlib.pyplot as plt

# Create a histogram
plt.hist(clv_values, bins=30, density=True, color="#FF5733", edgecolor="white"
plt.title("CLV Histogram")
plt.xlabel("CLV")
plt.ylabel("Density")
plt.show()
```



We get to know these things:

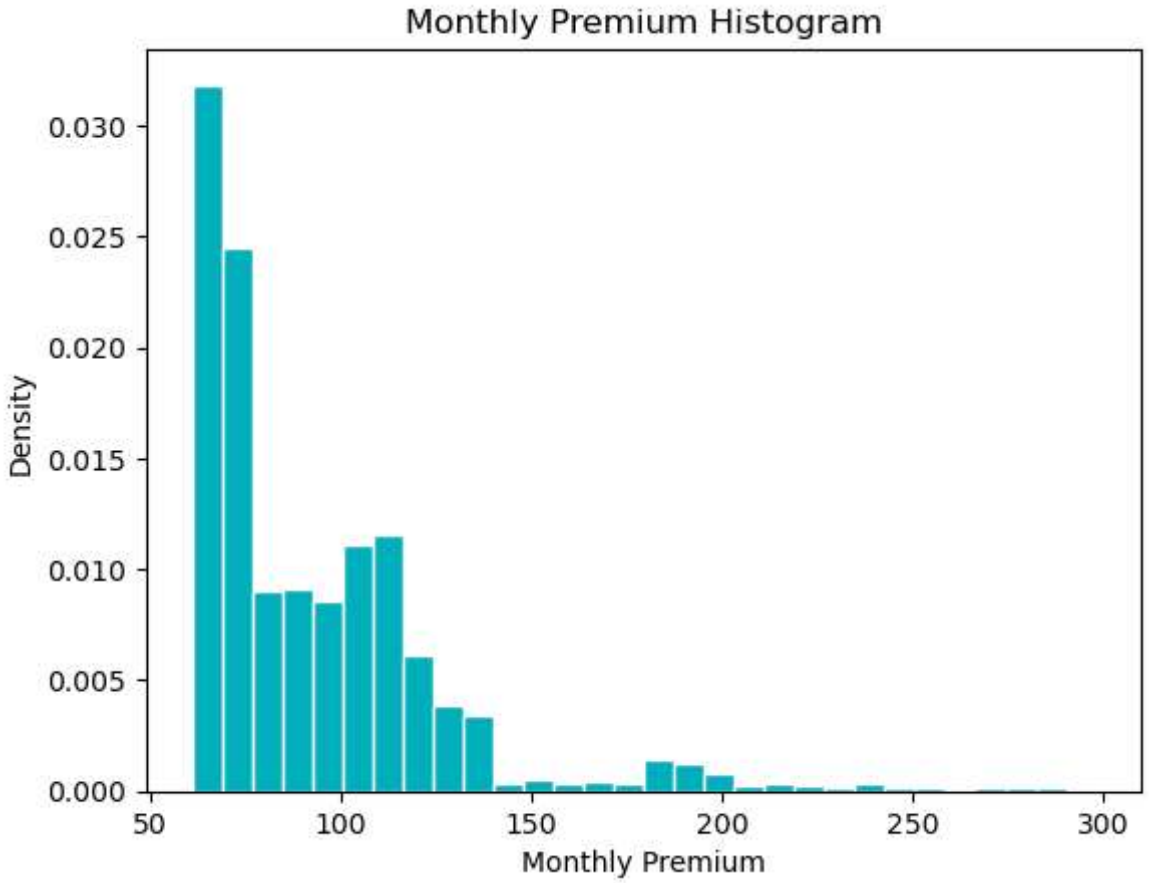
1. Maximum CLV is 83325.381 and the minimum CLV is 1898.008.
2. Mean of CLV is 8005 and the Median is 5780.
3. The Variance in CLV is 47210196 and the Standard Deviation is 687
0.968.
4. Skewness is 4.031284. CLV is positive skewed and most values are c

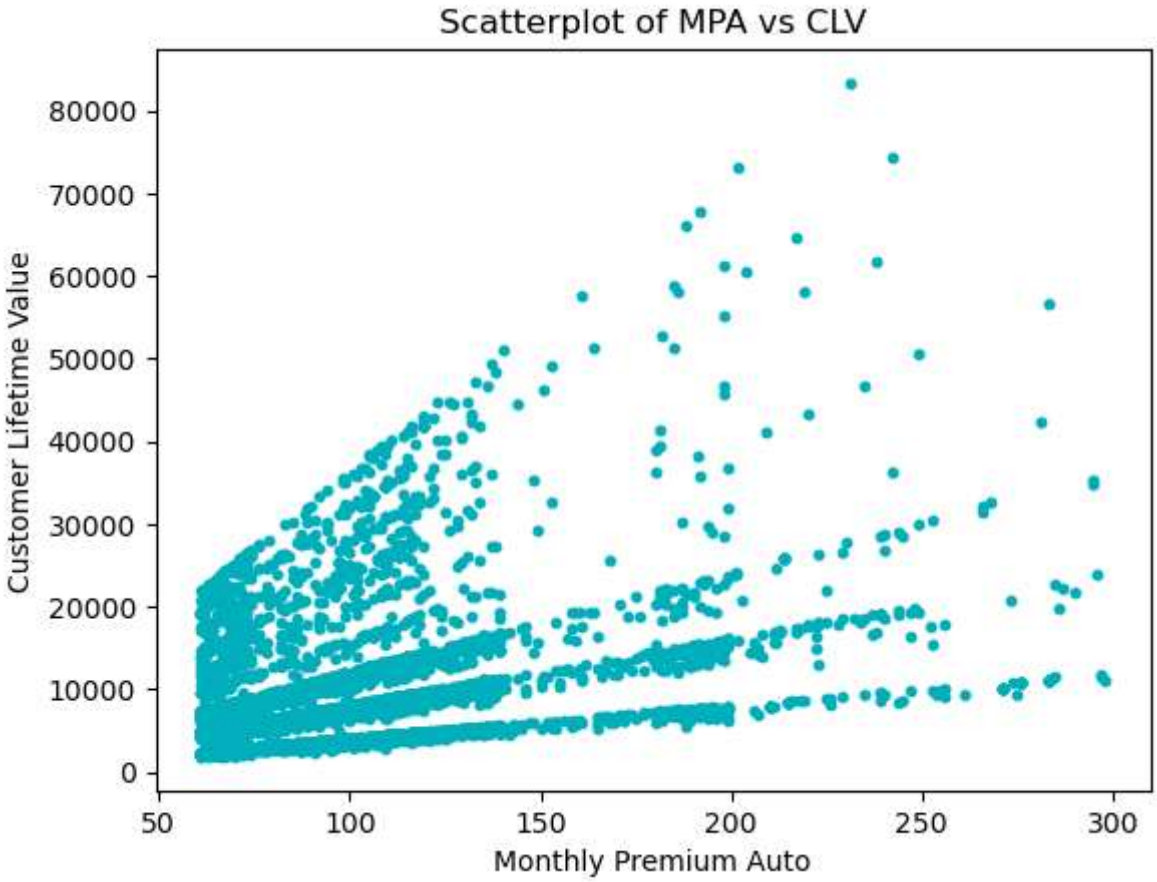
DESCRIPTIVE ANALYSIS of Monthly Premium Auto (MPA)

In [15]:
```
# Calculate the range, mean, standard deviation, summary statistics
mpa_values = InsuranceData['Monthly Premium Auto']

mpa_range = np.ptp(mpa_values)
mpa_mean = np.mean(mpa_values)
mpa_sd = np.std(mpa_values)
mpa_summary = mpa_values.describe()

# Calculate variance, skewness, and kurtosis
mpa_variance = np.var(mpa_values)
mpa_skewness = mpa_values.skew()
mpa_kurtosis = mpa_values.kurtosis()

# Calculate correlation with CustomerLifetimeValue
clv_values = InsuranceData['Customer Lifetime Value']
correlation = mpa_values.corr(clv_values)

# Print the results
print("Range:", mpa_range)
print("Mean:", mpa_mean)
print("Standard Deviation:", mpa_sd)
print("Summary Statistics:")
print(mpa_summary)
print("Variance:", mpa_variance)
print("Skewness:", mpa_skewness)
print("Kurtosis:", mpa_kurtosis)
print("Correlation with CustomerLifetimeValue:", correlation)
```

```
Range: 237
Mean: 93.21929056273265
Standard Deviation: 34.40608380986521
Summary Statistics:
count    9134.000000
mean       93.219291
std        34.407967
min        61.000000
25%        68.000000
50%        83.000000
75%       109.000000
max       298.000000
Name: Monthly Premium Auto, dtype: float64
Variance: 1183.7786031314686
Skewness: 2.1235464499475345
Kurtosis: 6.193604965840608
Correlation with CustomerLifetimeValue: 0.3962617375340209
```

In [16]:
```
# Create a histogram
plt.hist(mpa_values, bins=30, density=True, color="#00AFBB", edgecolor="white"
plt.title("Monthly Premium Histogram")
plt.xlabel("Monthly Premium")
plt.ylabel("Density")
plt.show()

# Create a scatterplot
plt.scatter(x=mpa_values, y=clv_values, color="#00AFBB", s=10)
plt.xlabel("Monthly Premium Auto")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of MPA vs CLV")
plt.show()
```





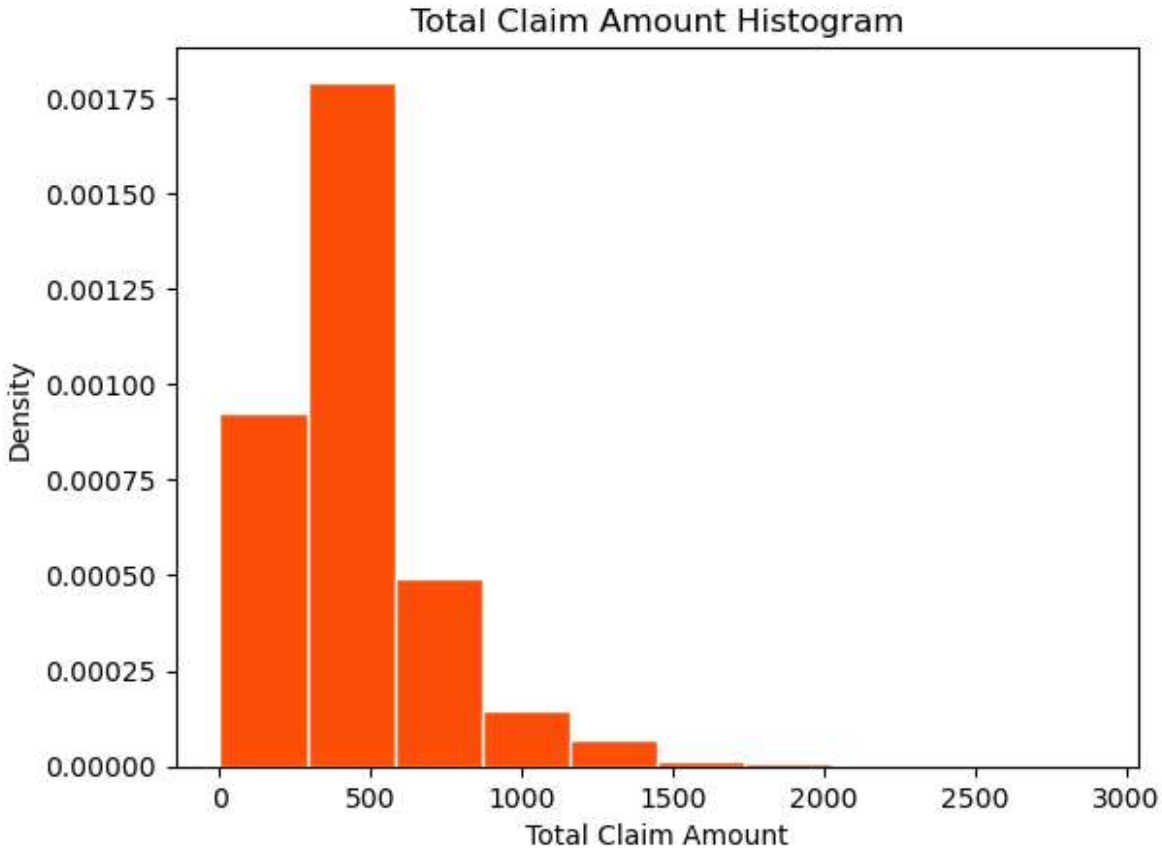
WE GET TO KNOW THESE THINGS :

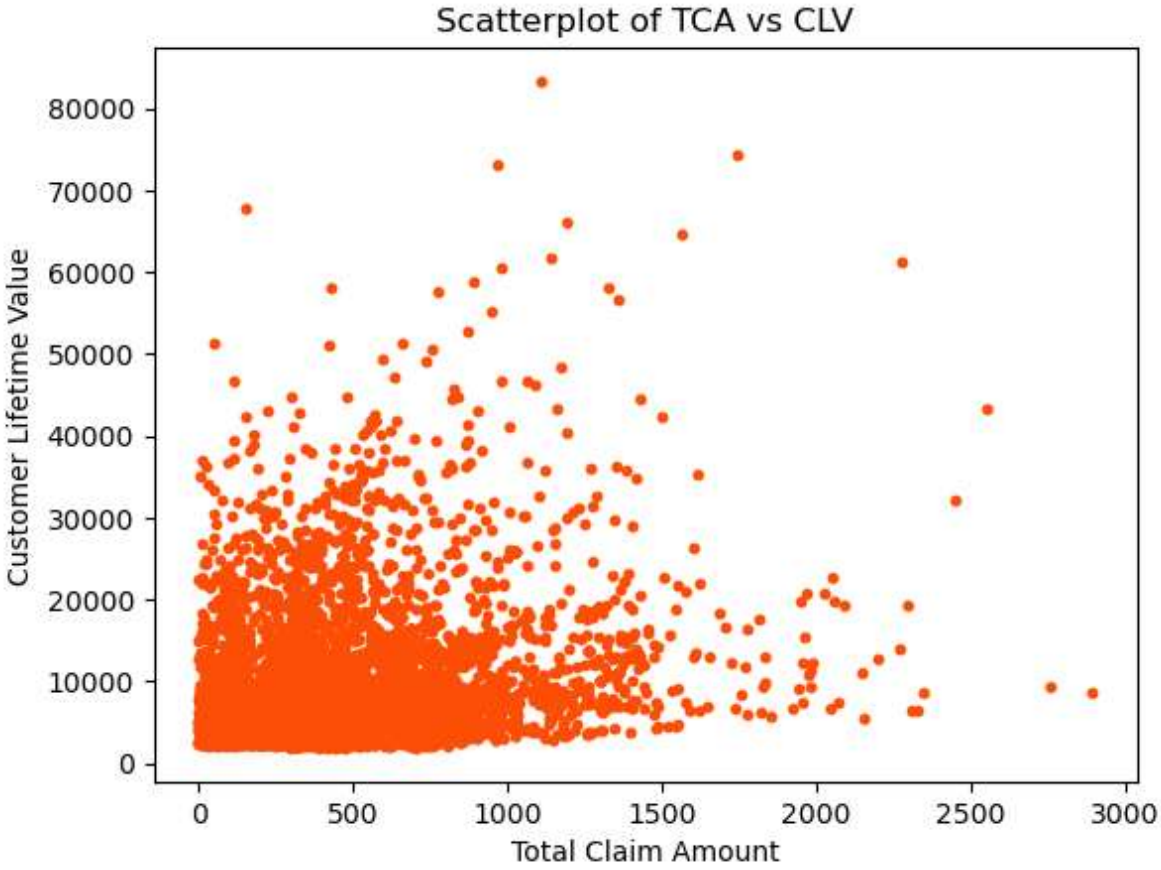
1. Maximum MPA is 298 and the minimum MPA is 61
2. Mean of MPA is 93.21929 and the Median is 84.00
3. The Variance in MPA is 1183.908 and the Standard Deviation is 34.40797
4. Skewness is 2.122849. MPA is positive skewed and most values are concentrated on the left of
    the mean value, yet all the extreme values are on the right of the mean value.
5. Kurtosis is 6.187546. Since kurtosis > 3, means distribution has thicker tails than normal
    distribution and have more outliers (extreme values).
6. There is a Positive Corelation of 39.62 % of MPA with CLV. From scatter plot, it is clearly
    visible that on MPA, CLV is also Increasing.7.
7. Monthly premiums follow a trend similar to CLV although the distribution is NOT as skewed
    or as long tailed as CLV. This can be visually seen in the Histogram.

DESCRPTIVE ANALYSIS OF TOTAL CLAIM AMOUNT (TCA)

In [17]:
```
# Calculate the range, mean, standard deviation, summary statistics
tca_values = InsuranceData['Total Claim Amount']

tca_range = np.ptp(tca_values)
tca_mean = np.mean(tca_values)
tca_sd = np.std(tca_values)
tca_summary = tca_values.describe()

# Calculate variance, skewness, and kurtosis
tca_variance = np.var(tca_values)
tca_skewness = tca_values.skew()
tca_kurtosis = tca_values.kurtosis()

# Calculate correlation with CustomerLifetimeValue
clv_values = InsuranceData['Customer Lifetime Value']
correlation = tca_values.corr(clv_values)

# Print the results
print("Range:", tca_range)
print("Mean:", tca_mean)
print("Standard Deviation:", tca_sd)
print("Summary Statistics:")
print(tca_summary)
print("Variance:", tca_variance)
print("Skewness:", tca_skewness)
print("Kurtosis:", tca_kurtosis)
print("Correlation with CustomerLifetimeValue:", correlation)

# Create a histogram
plt.hist(tca_values, bins=10, density=True, color="#FC4E07", edgecolor="white"
plt.title("Total Claim Amount Histogram")
plt.xlabel("Total Claim Amount")
plt.ylabel("Density")
plt.show()

# Create a scatterplot
plt.scatter(x=tca_values, y=clv_values, color="#FC4E07", s=10)
plt.xlabel("Total Claim Amount")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of TCA vs CLV")
plt.show()
```

```
Range: 2893.140671
Mean: 434.0887943128969
Standard Deviation: 290.48418923888653
Summary Statistics:
count    9134.000000
mean      434.088794
std       290.500092
min         0.099007
25%       272.258244
50%       383.945434
75%       547.514839
max      2893.239678
Name: Total Claim Amount, dtype: float64
Variance: 84381.06419777324
Skewness: 1.7149658097209557
Kurtosis: 5.979401045843765
Correlation with CustomerLifetimeValue: 0.22645091528641698
```





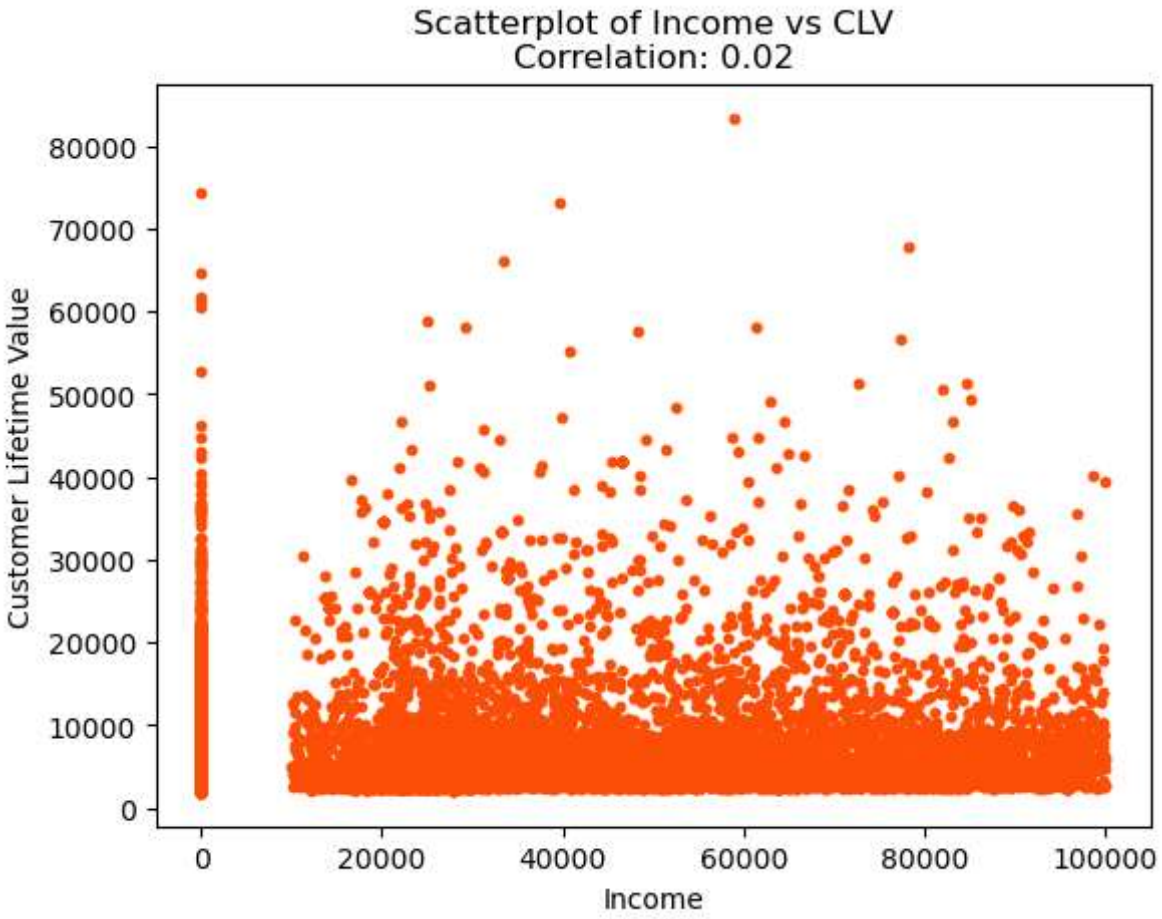
WE GET TO KNOW THESE FOLLOWING THINGS:

1. Maximum TCA is 0.099007 and the minimum TCA is 2893.239678
2. Mean of TCA is 434.0888 and the Median is 383.945
3. The Variance in TCA is 84390.3 and the Standard Deviation is 290.5001
4. Skewness is 1.714403. TCA is positive skewed and most values are concentrated on the left of
the mean value, yet all the extreme values are on the right of the mean value.
5. Kurtosis is 5.973506. Since kurtosis > 3, means TCA distribution has thicker tails than normal
distribution and have more outliers (extreme values).
6. There is a Positive Corelation of 22.65 % of TCA with CLV. From scatter plot, it is clearly
visible that on TCA, CLV is also Increasing.
7. Total Claim amounts also follow a trend similar to CLV and MPA although the distribution is
8. NOT as skewed or as long tailed as MPA. This can be visually seen in the Histogram.

This means that variation in data is CLV > MPA > TCA

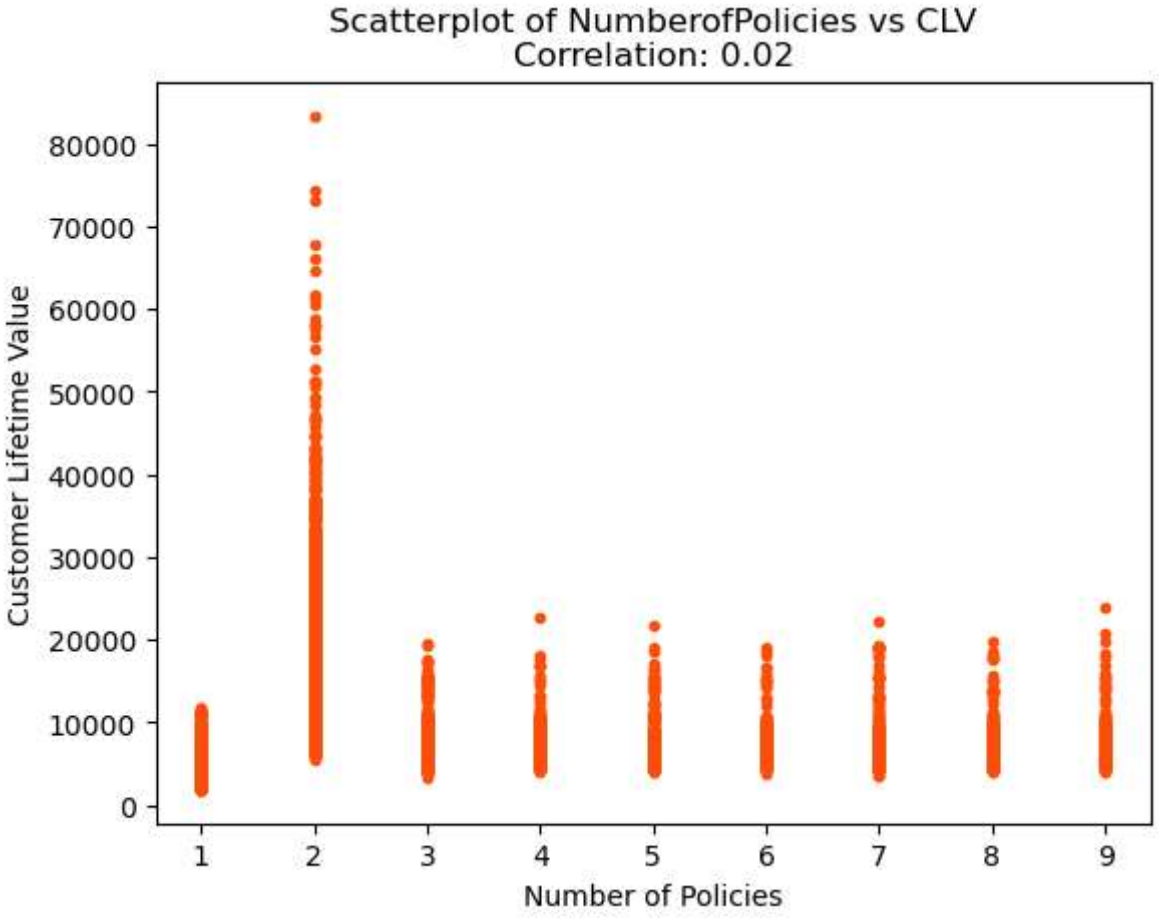
Descriptive Analysis of other variables:

In [18]:
```
import matplotlib.pyplot as plt

# Calculate correlation and create scatterplot for Income vs CLV
cor_income_clv = InsuranceData['Income'].corr(InsuranceData['Customer Lifetime
plt.scatter(x=InsuranceData['Income'], y=InsuranceData['Customer Lifetime Valu
plt.xlabel("Income")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of Income vs CLV\nCorrelation: {:.2f}".format(cor_incom
plt.show()
```

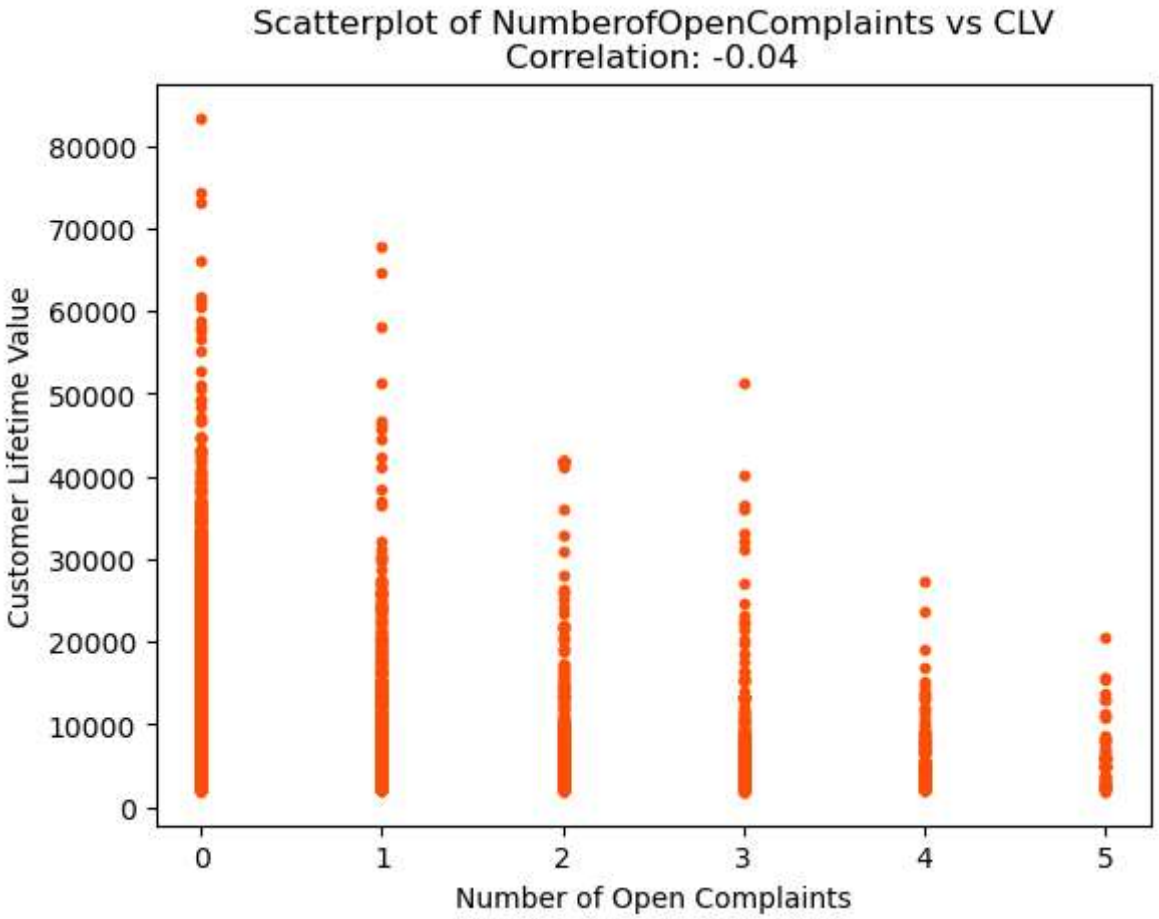


In [19]:

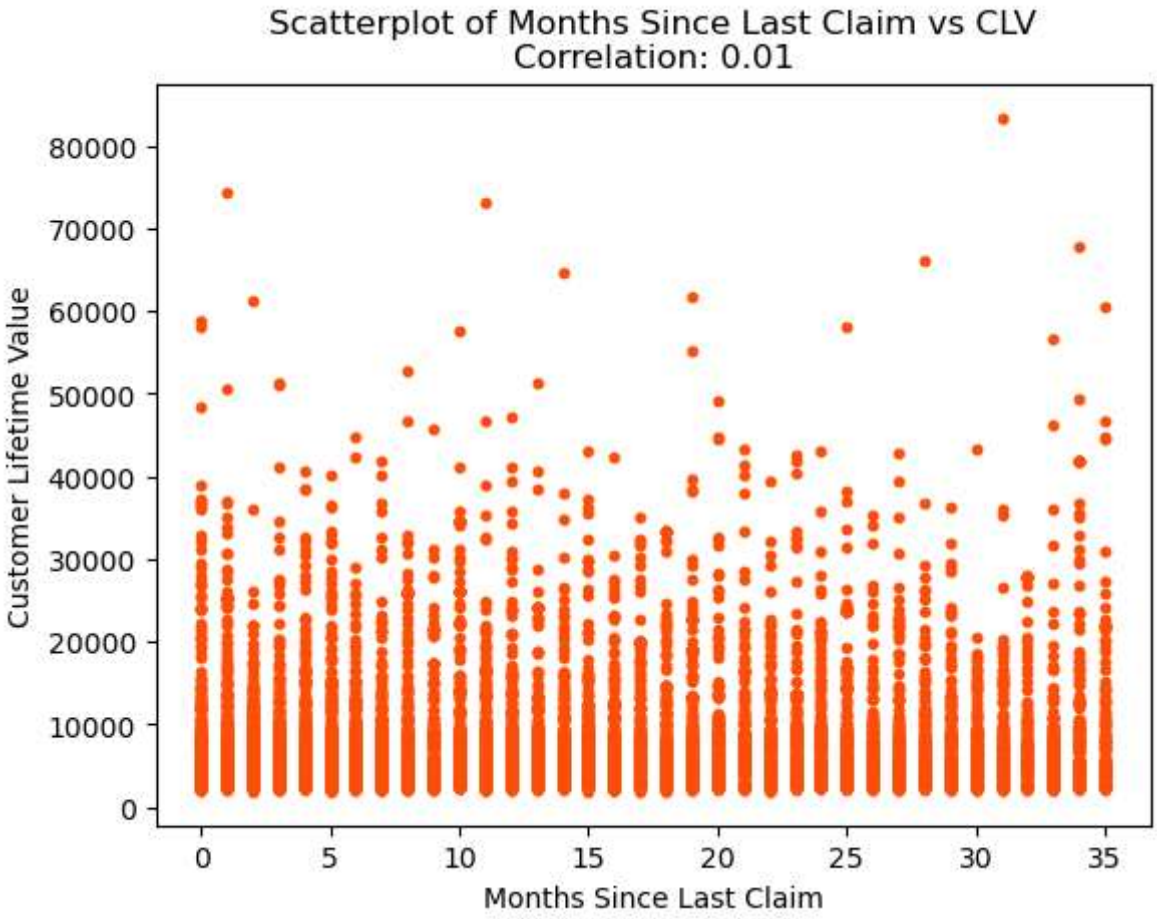
```
# Calculate correlation and create scatterplot for NumberofPolicies vs CLV
cor_nop_clv = InsuranceData['Number of Policies'].corr(InsuranceData['Customer
plt.scatter(x=InsuranceData['Number of Policies'], y=InsuranceData['Customer L
plt.xlabel("Number of Policies")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of NumberofPolicies vs CLV\nCorrelation: {:.2f}".format
plt.show()
```



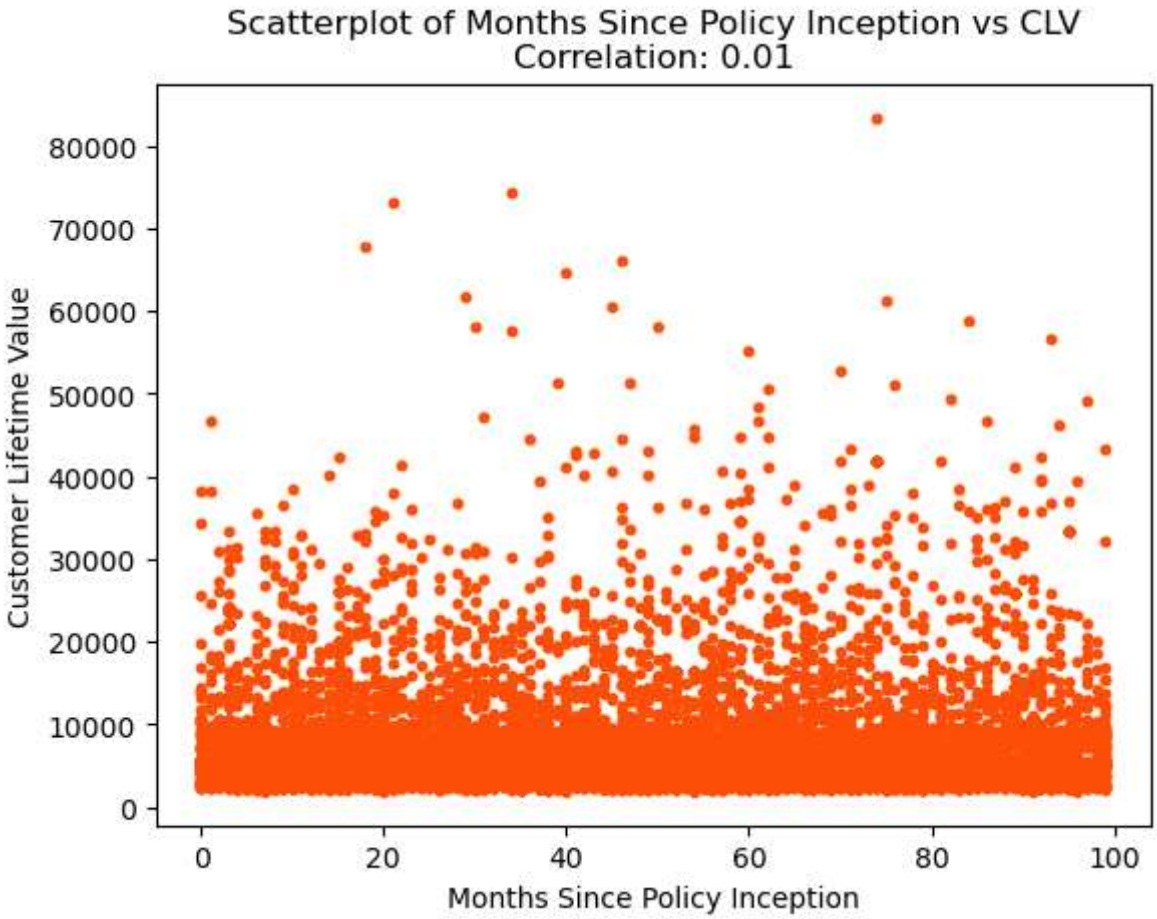
In [20]:
```
# Calculate correlation and create scatterplot for NumberofOpenComplaints vs C
cor_noc_clv = InsuranceData['Number of Open Complaints'].corr(InsuranceData['C
plt.scatter(x=InsuranceData['Number of Open Complaints'], y=InsuranceData['Cus
plt.xlabel("Number of Open Complaints")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of NumberofOpenComplaints vs CLV\nCorrelation: {:.2f}".
plt.show()
```



In [21]:
```
# Calculate correlation and create scatterplot for MonthsSinceLastClaim vs CLV
cor_mslc_clv = InsuranceData['Months Since Last Claim'].corr(InsuranceData['Cu
plt.scatter(x=InsuranceData['Months Since Last Claim'], y=InsuranceData['Custo
plt.xlabel("Months Since Last Claim")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of Months Since Last Claim vs CLV\nCorrelation: {:.2f}"
plt.show()
```



In [22]:
```
# Calculate correlation and create scatterplot for MonthsSincePolicyInception
cor_mspi_clv = InsuranceData['Months Since Policy Inception'].corr(InsuranceDa
plt.scatter(x=InsuranceData['Months Since Policy Inception'], y=InsuranceData[
plt.xlabel("Months Since Policy Inception")
plt.ylabel("Customer Lifetime Value")
plt.title("Scatterplot of Months Since Policy Inception vs CLV\nCorrelation: {
plt.show()
```

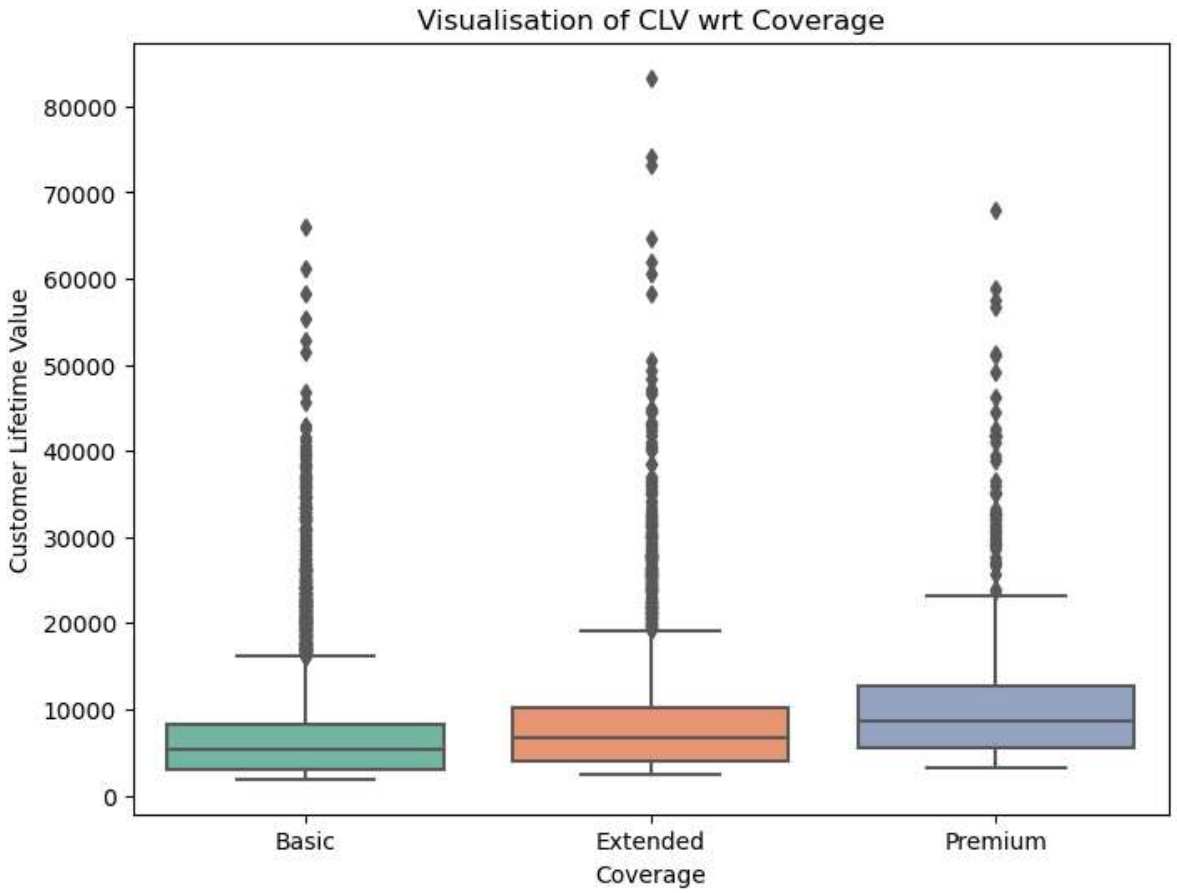


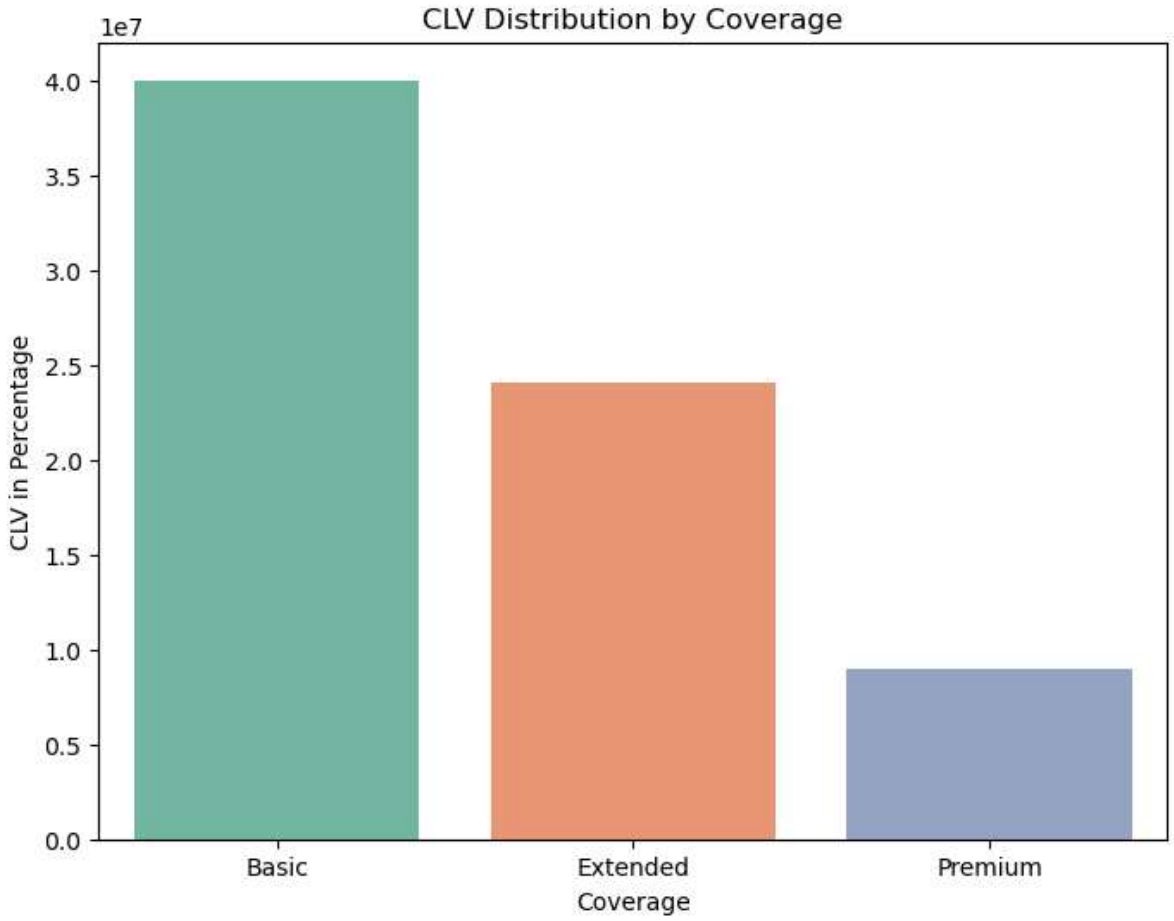
The positive correlation values close to zero show that that there is no strong relationship of Income, Months Since Last Claim, Number of Policies etc with CLV.

INFERENTIAL STATISTICS

Effect of Insurance Coverage on Customer Life Time Value (CLV)

In [21]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot: Visualisation of CLV wrt Coverage
plt.figure(figsize=(8, 6))
sns.boxplot(x='Coverage', y='Customer Lifetime Value', data=InsuranceData, pal
plt.xlabel('Coverage')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Coverage')
plt.show()

# Bar plot: CLV Distribution by Coverage
aggData = InsuranceData.groupby('Coverage')['Customer Lifetime Value'].sum().r

plt.figure(figsize=(8, 6))
sns.barplot(x='Coverage', y='Customer Lifetime Value', data=aggData, palette='
plt.xlabel('Coverage')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Coverage')
plt.show()
```

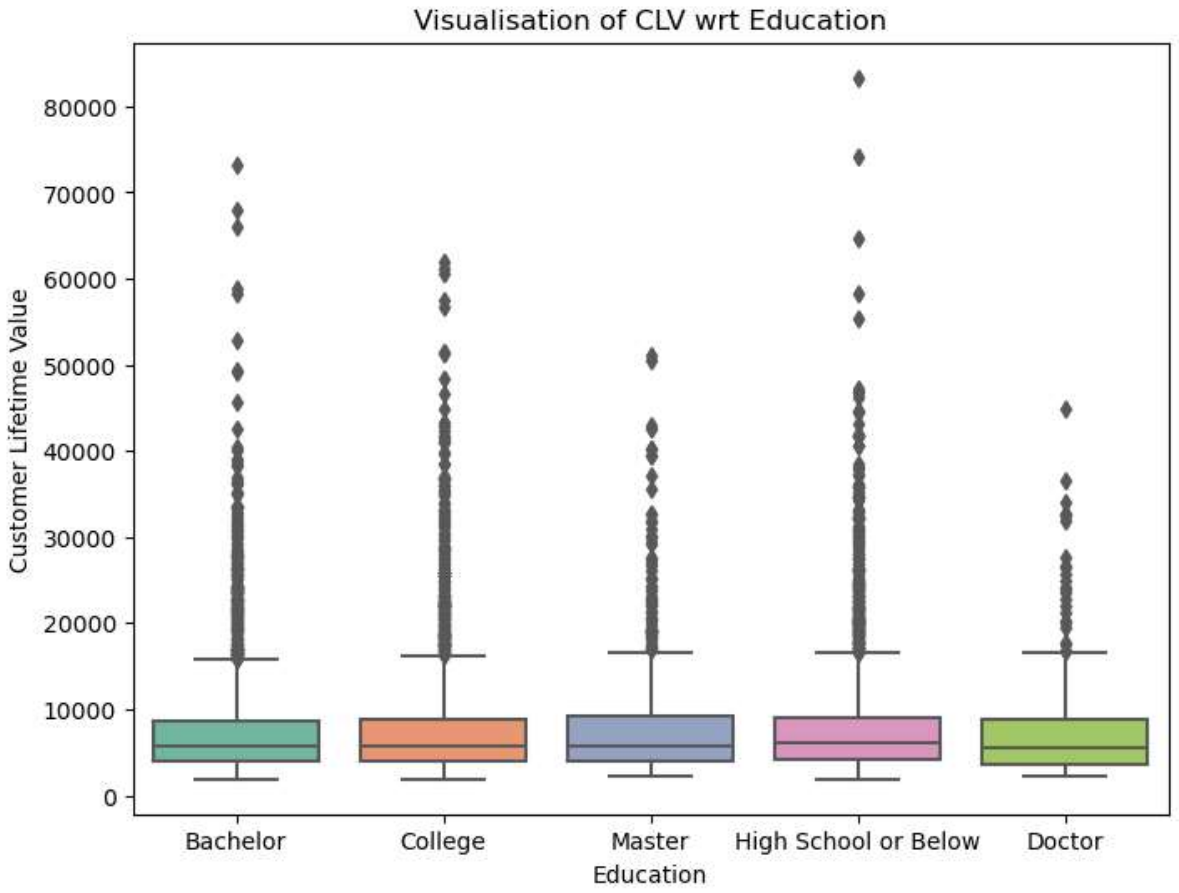


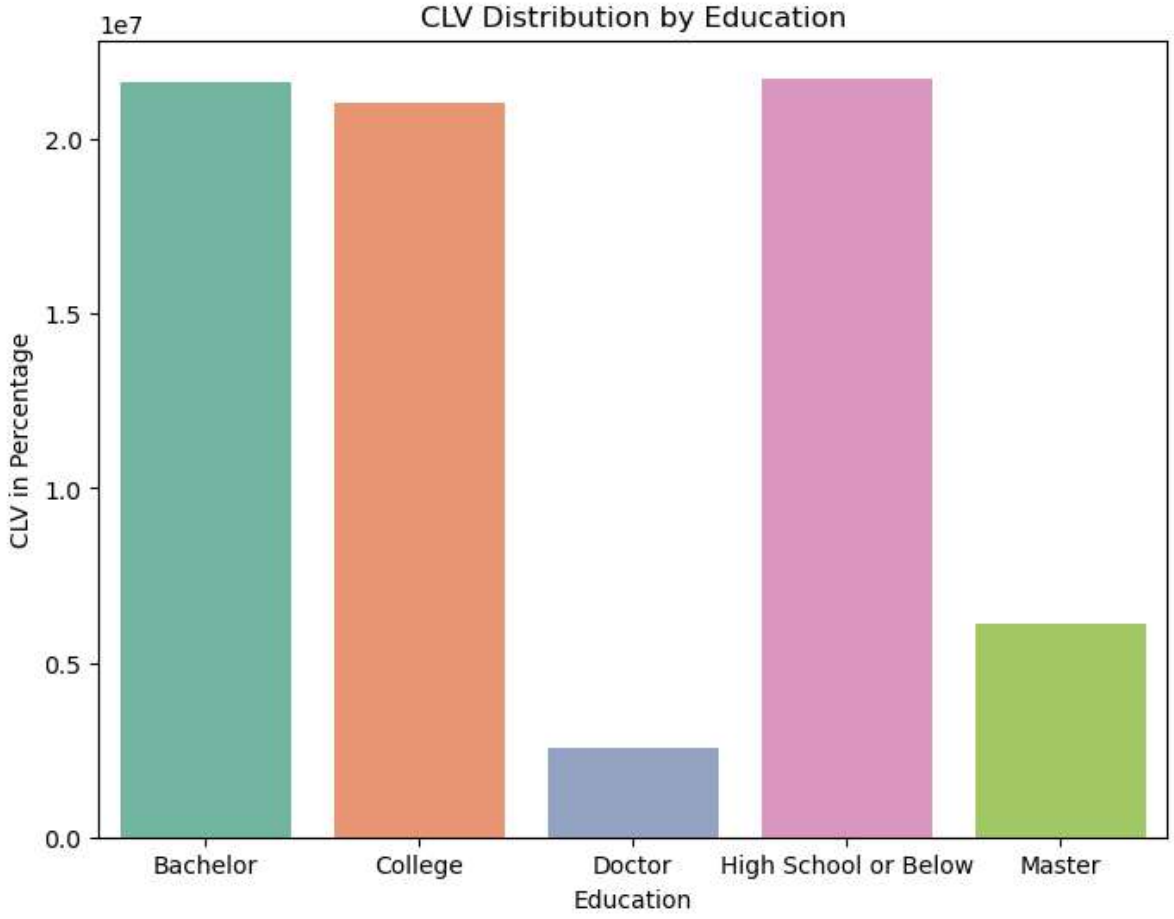


Customers who have taken Basic Insurance for their vehicals are more valuable then Extended or Premium Insurance Policy holders.

Effect of Education on Customer Life Time Value (CLV)

In [24]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot: Visualisation of CLV wrt Education
plt.figure(figsize=(8, 6))
sns.boxplot(x='Education', y='Customer Lifetime Value', data=InsuranceData, pa
plt.xlabel('Education')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Education')
plt.show()

# Bar plot: CLV Distribution by Education
aggData = InsuranceData.groupby('Education')['Customer Lifetime Value'].sum().

plt.figure(figsize=(8, 6))
sns.barplot(x='Education', y='Customer Lifetime Value', data=aggData, palette=
plt.xlabel('Education')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Education')
plt.show()
```

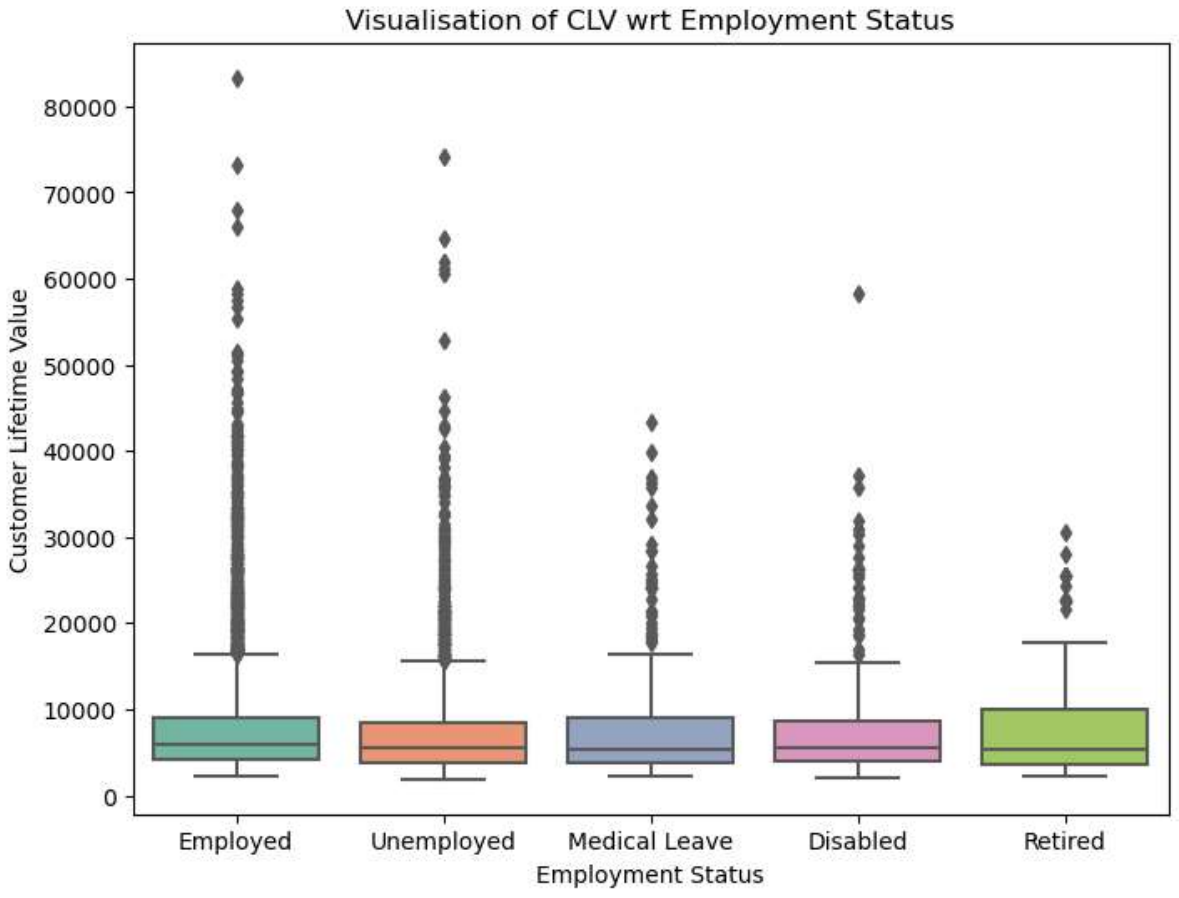


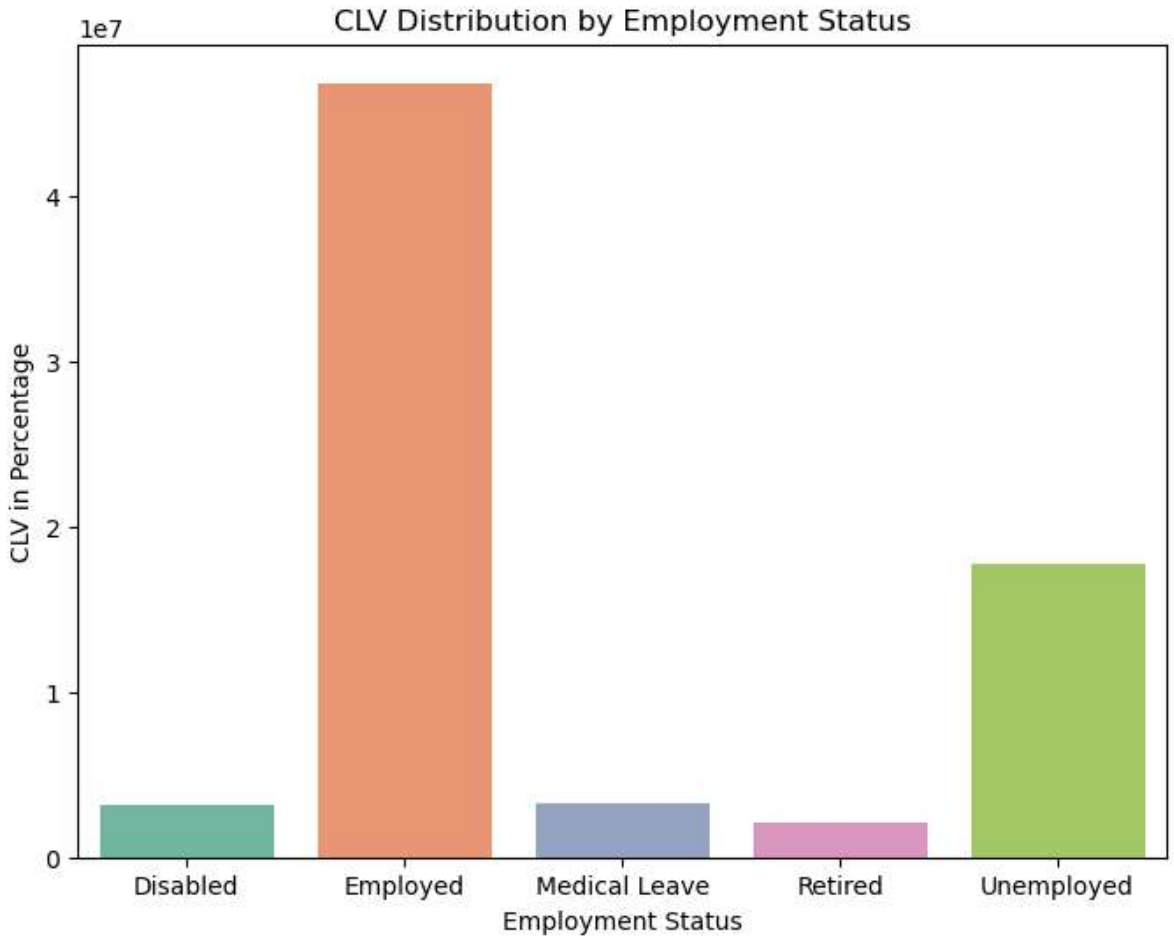


Educated customers (with a bachelors or equivalent degree) are more valuable than others.

Effect of Employment Status on Customer Life Time Value (CLV)

In [25]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot: Visualisation of CLV wrt Employment Status
plt.figure(figsize=(8, 6))
sns.boxplot(x='EmploymentStatus', y='Customer Lifetime Value', data=InsuranceD
plt.xlabel('Employment Status')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Employment Status')
plt.show()

# Bar plot: CLV Distribution by Employment Status
aggData = InsuranceData.groupby('EmploymentStatus')['Customer Lifetime Value']

plt.figure(figsize=(8, 6))
sns.barplot(x='EmploymentStatus', y='Customer Lifetime Value', data=aggData, p
plt.xlabel('Employment Status')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Employment Status')
plt.show()
```

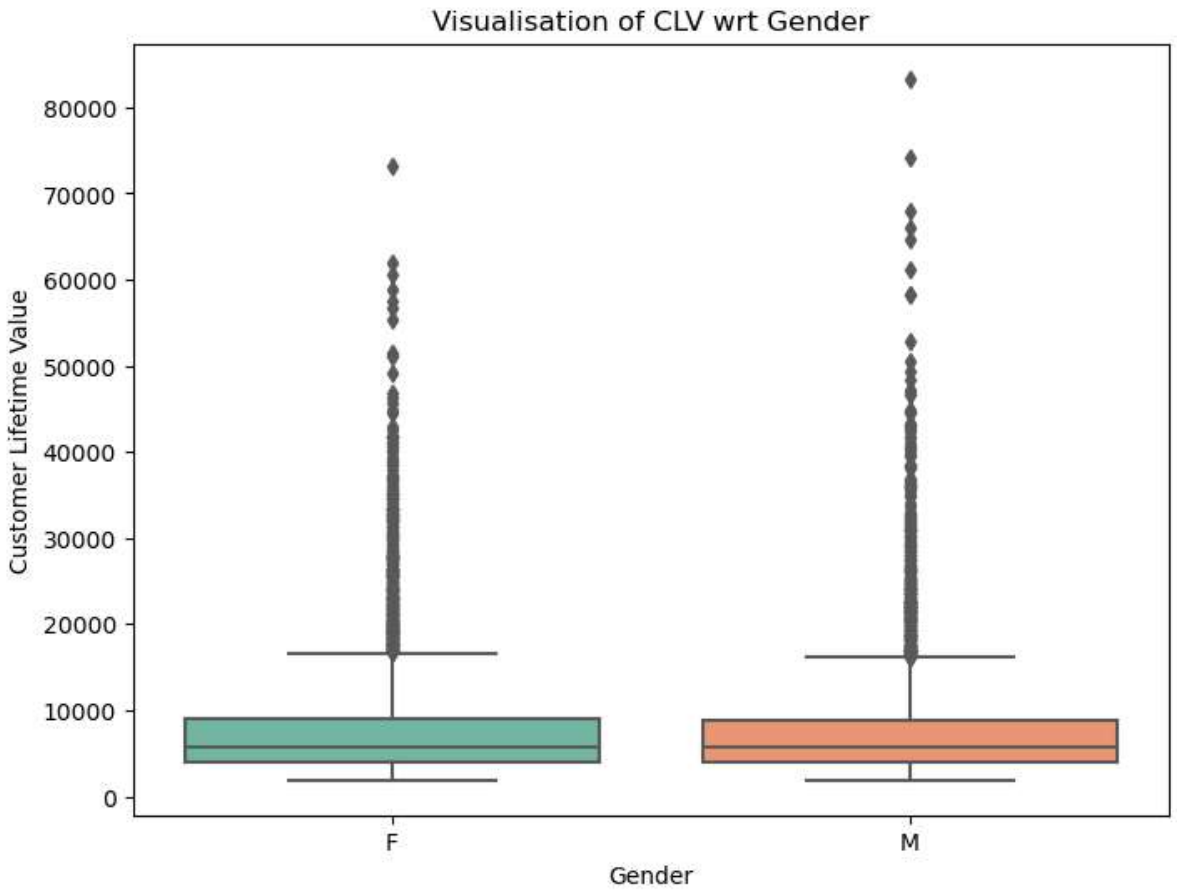


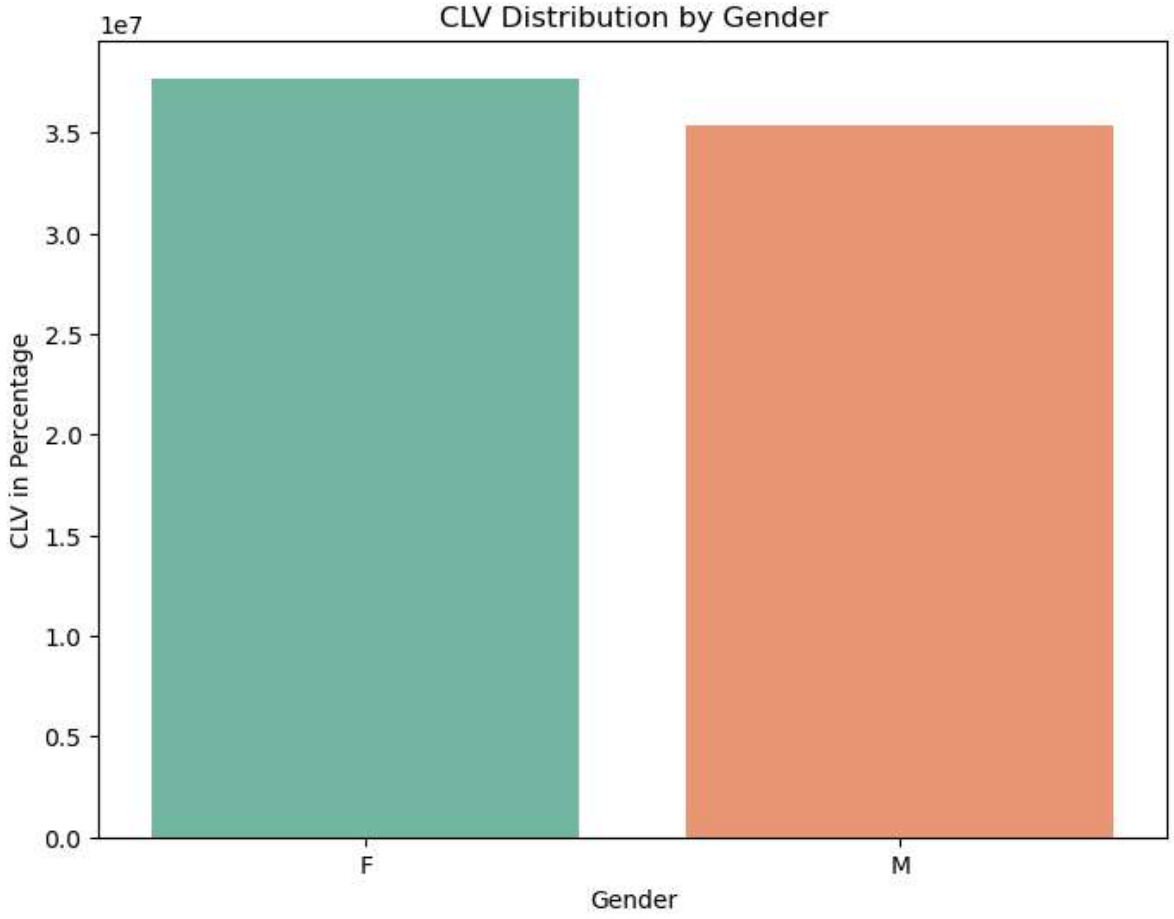


Employed customers are more valuable than others as compared to Retired, Unemployed or Disabled Customers.

Effect of Gender on Customer Life Time Value (CLV)

In [26]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot: Visualisation of CLV wrt Gender
plt.figure(figsize=(8, 6))
sns.boxplot(x='Gender', y='Customer Lifetime Value', data=InsuranceData, palet
plt.xlabel('Gender')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Gender')
plt.show()

# Bar plot: CLV Distribution by Gender
aggData = InsuranceData.groupby('Gender')['Customer Lifetime Value'].sum().res

plt.figure(figsize=(8, 6))
sns.barplot(x='Gender', y='Customer Lifetime Value', data=aggData, palette='Se
plt.xlabel('Gender')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Gender')
plt.show()
```

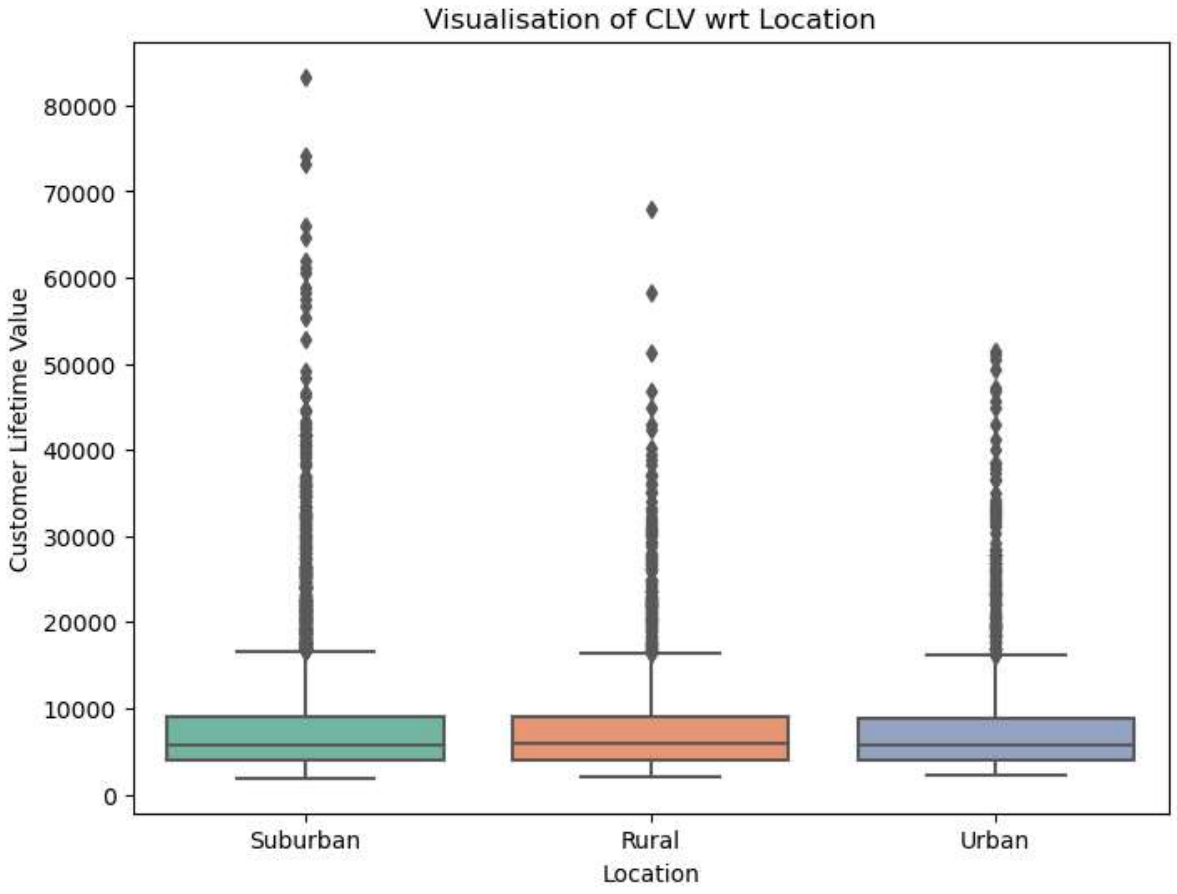


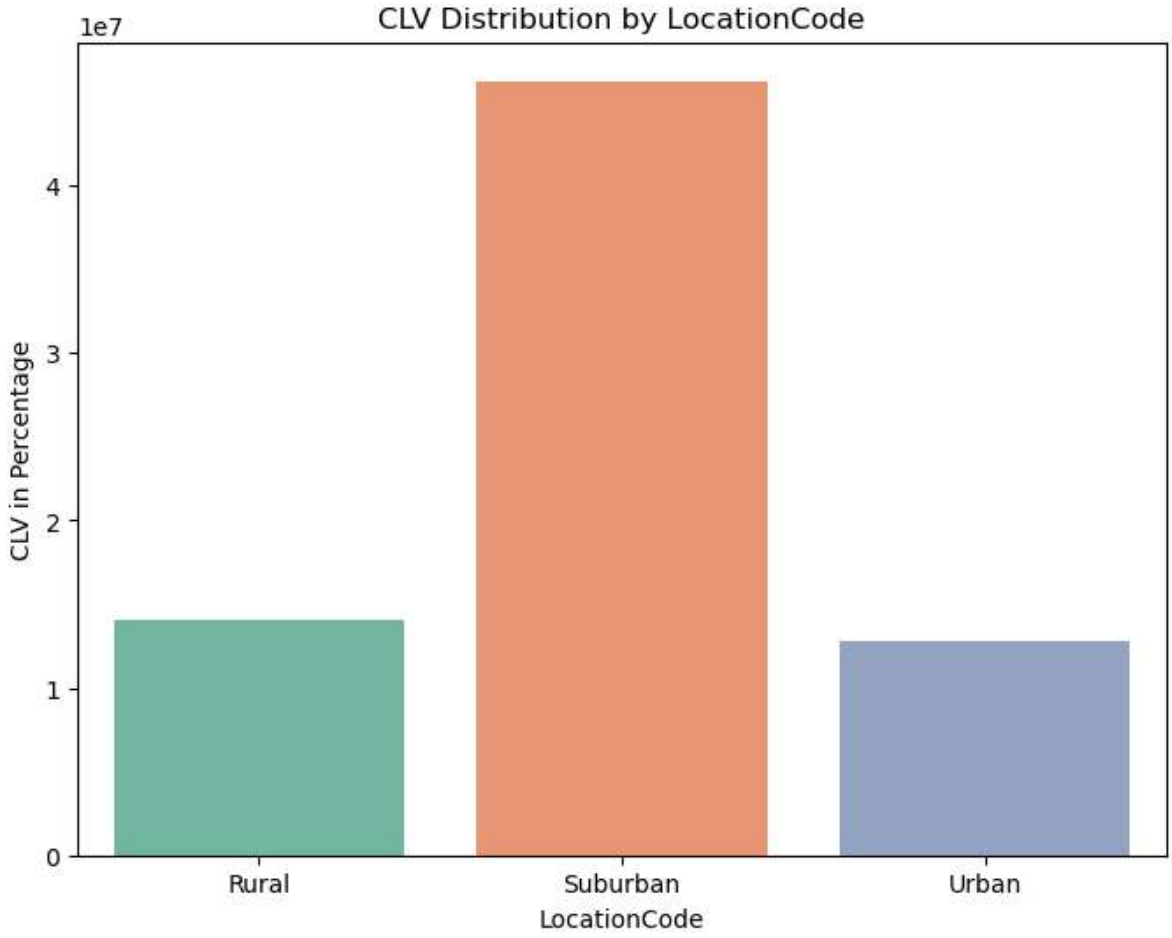


Gender has no role to play in determining the value of a customer. Both Male and Female looks valuable.

Effect of Location on Customer Life Time Value (CLV)

In [27]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot: Visualisation of CLV wrt Location
plt.figure(figsize=(8, 6))
sns.boxplot(x='Location Code', y='Customer Lifetime Value', data=InsuranceData
plt.xlabel('Location')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Location')
plt.show()

# Bar plot: CLV Distribution by Location
aggData = InsuranceData.groupby('Location Code')['Customer Lifetime Value'].su

plt.figure(figsize=(8, 6))
sns.barplot(x='Location Code', y='Customer Lifetime Value', data=aggData, pale
plt.xlabel('LocationCode')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by LocationCode')
plt.show()
```

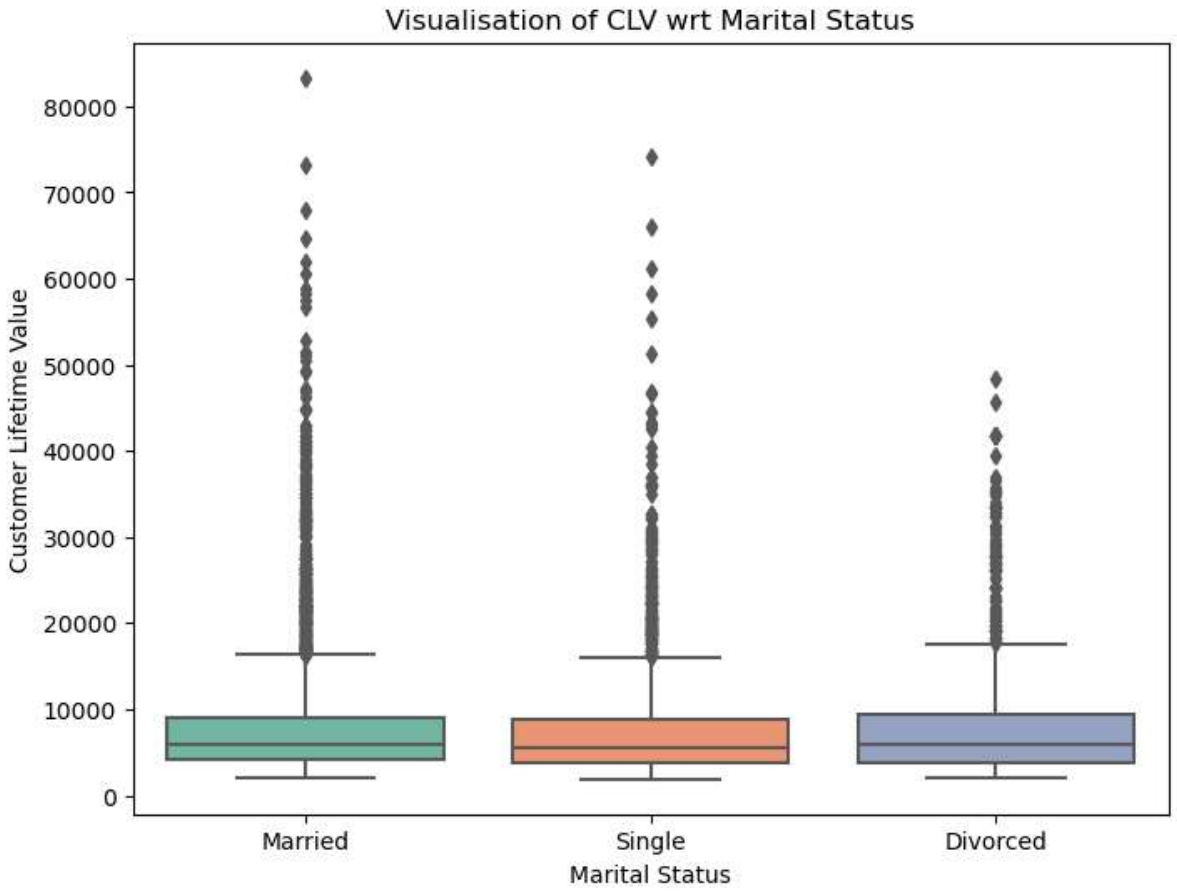


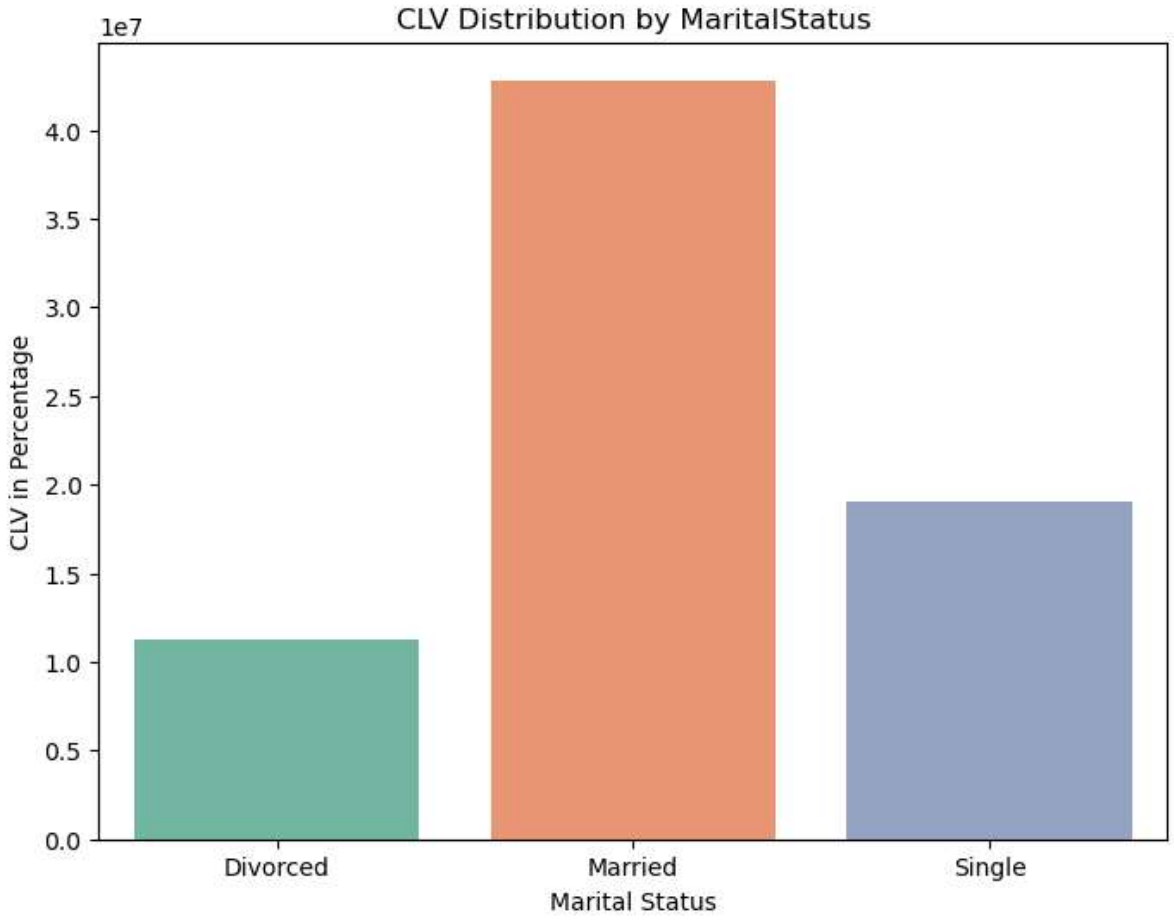


Rural customers are LESS valuable than Urban customers.

Effect of Marital Status on Customer Life Time Value (CLV)

In [28]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot: Visualisation of CLV wrt Marital Status
plt.figure(figsize=(8, 6))
sns.boxplot(x='Marital Status', y='Customer Lifetime Value', data=InsuranceDat
plt.xlabel('Marital Status')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Marital Status')
plt.show()

# Bar plot: CLV Distribution by Marital Status
aggData = InsuranceData.groupby('Marital Status')['Customer Lifetime Value'].s

plt.figure(figsize=(8, 6))
sns.barplot(x='Marital Status', y='Customer Lifetime Value', data=aggData, pal
plt.xlabel('Marital Status')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by MaritalStatus')
plt.show()
```

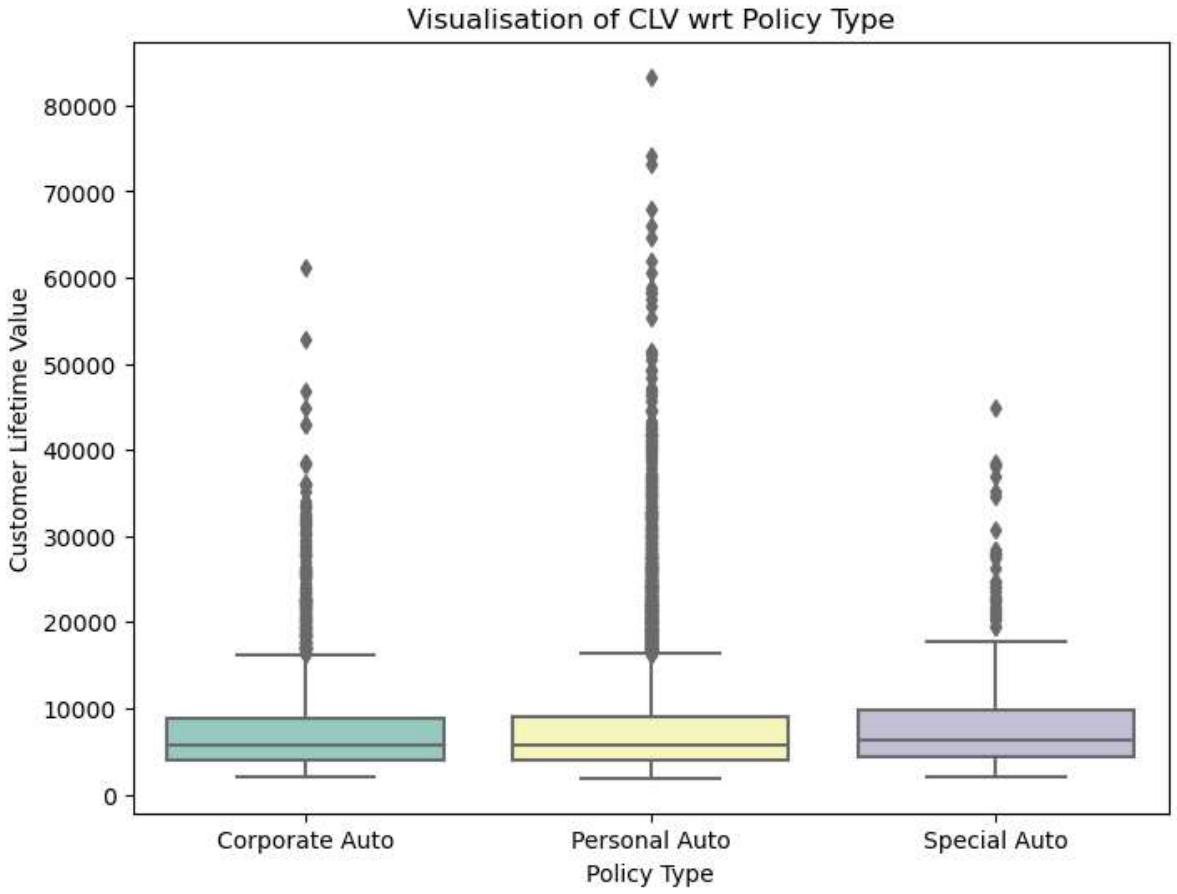


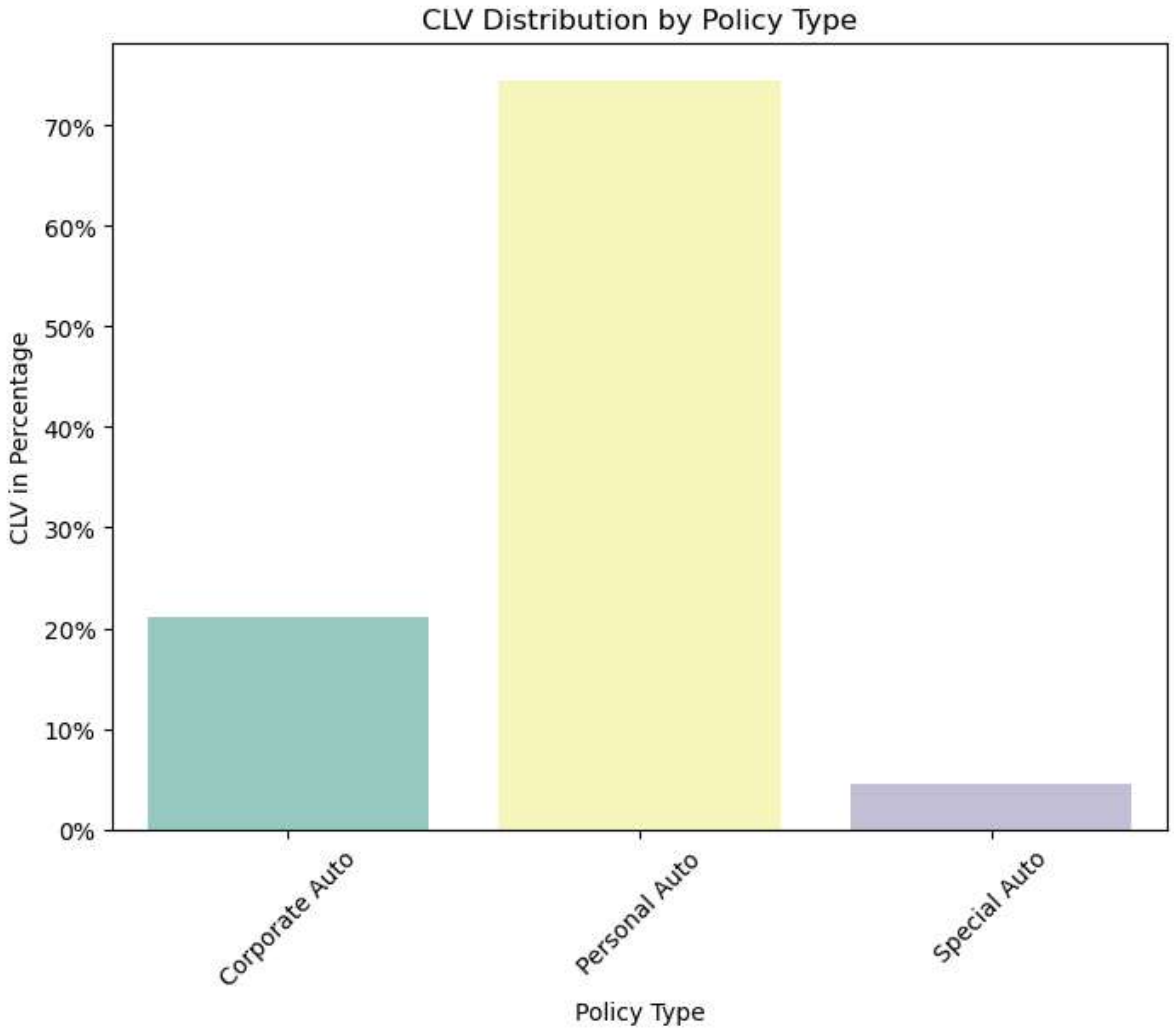


Married customers are buying more auto insurance and adding more value to company.

Effect of Policy Type on Customer Life Time Value (CLV)

In [29]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='Policy Type', y='Customer Lifetime Value', data=InsuranceData,
plt.xlabel('Policy Type')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Policy Type')
plt.show()

# Bar chart
aggData = InsuranceData.groupby('Policy Type')['Customer Lifetime Value'].sum(

plt.figure(figsize=(8, 6))
sns.barplot(x='Policy Type', y=aggData['Customer Lifetime Value'] / aggData['C
plt.xlabel('Policy Type')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Policy Type')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```

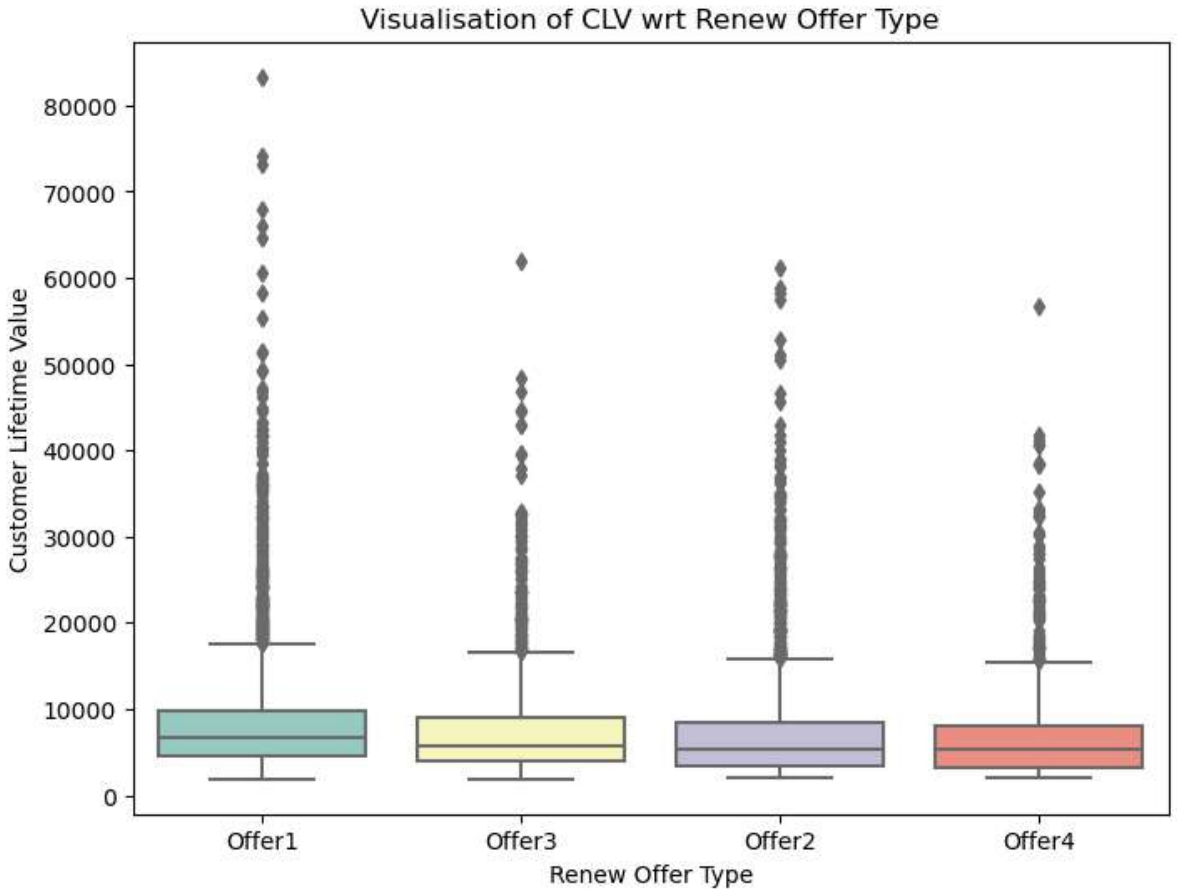


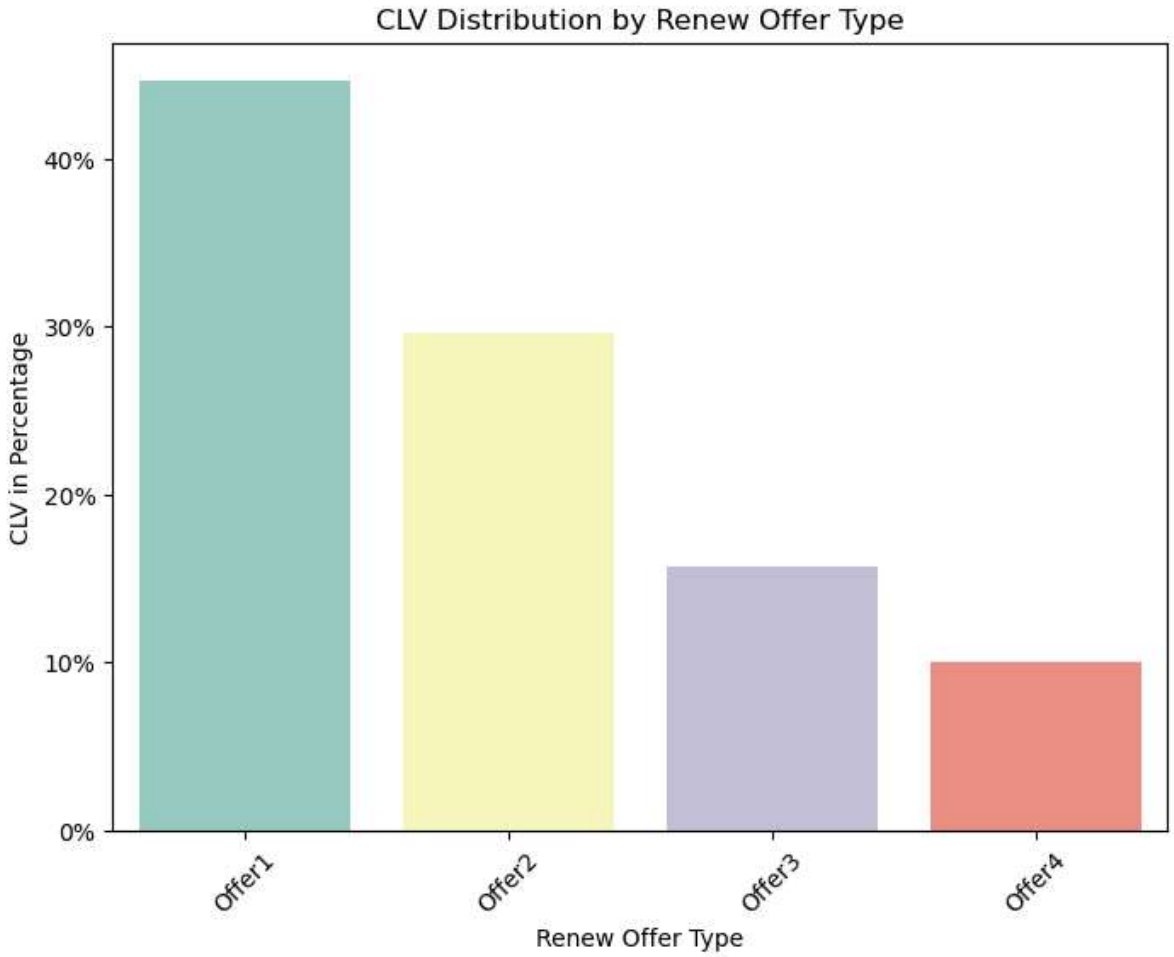


Customers having their own Persoanl Policy are more valuable to company then Corporate and Special Insurance policy holder.

Effect of Renew Offer Type on Customer Life Time Value (CLV)

In [30]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='Renew Offer Type', y='Customer Lifetime Value', data=InsuranceD
plt.xlabel('Renew Offer Type')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Renew Offer Type')
plt.show()

# Bar chart
aggData = InsuranceData.groupby('Renew Offer Type')['Customer Lifetime Value']

plt.figure(figsize=(8, 6))
sns.barplot(x='Renew Offer Type', y=aggData['Customer Lifetime Value'] / aggDa
plt.xlabel('Renew Offer Type')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Renew Offer Type')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```



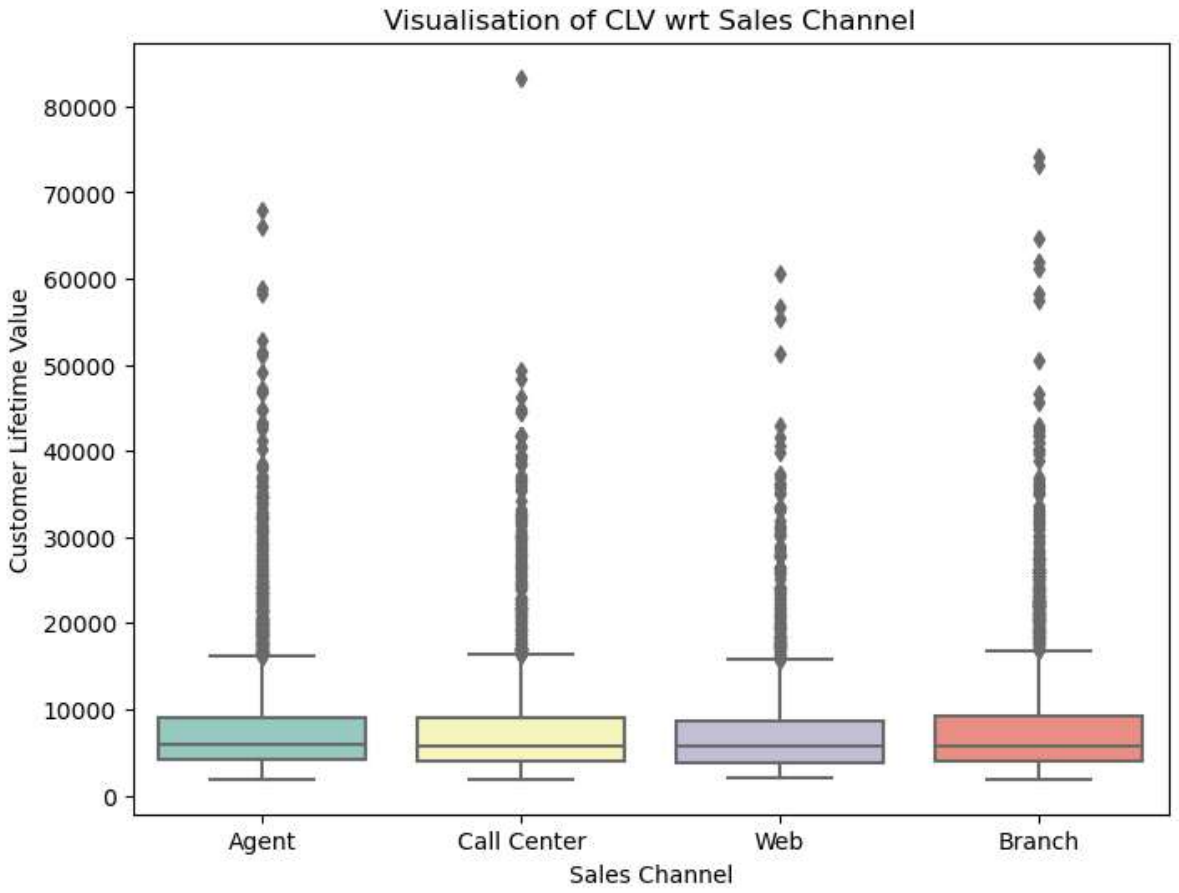


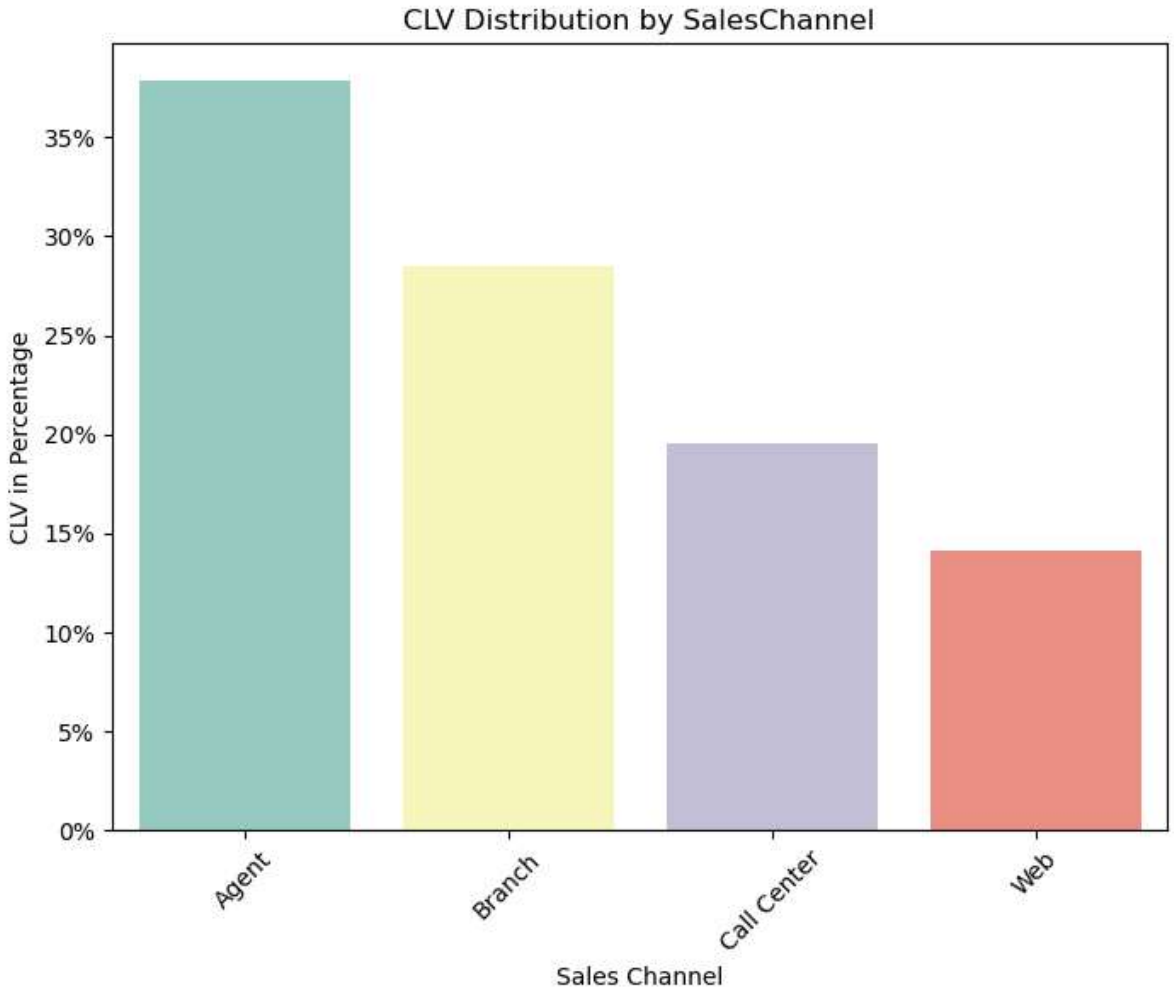
Offers 1 and Offer 2 attracts more customers.

Effect of Sales Channel on Customer Life Time Value (CLV)

In [31]:

```
# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='Sales Channel', y='Customer Lifetime Value', data=InsuranceData
plt.xlabel('Sales Channel')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Sales Channel')
plt.show()

# Bar chart
aggData = InsuranceData.groupby('Sales Channel')['Customer Lifetime Value'].su

plt.figure(figsize=(8, 6))
sns.barplot(x='Sales Channel', y=aggData['Customer Lifetime Value'] /
            aggData['Customer Lifetime Value'].sum(), data=aggData, palette='S
plt.xlabel('Sales Channel')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by SalesChannel')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```



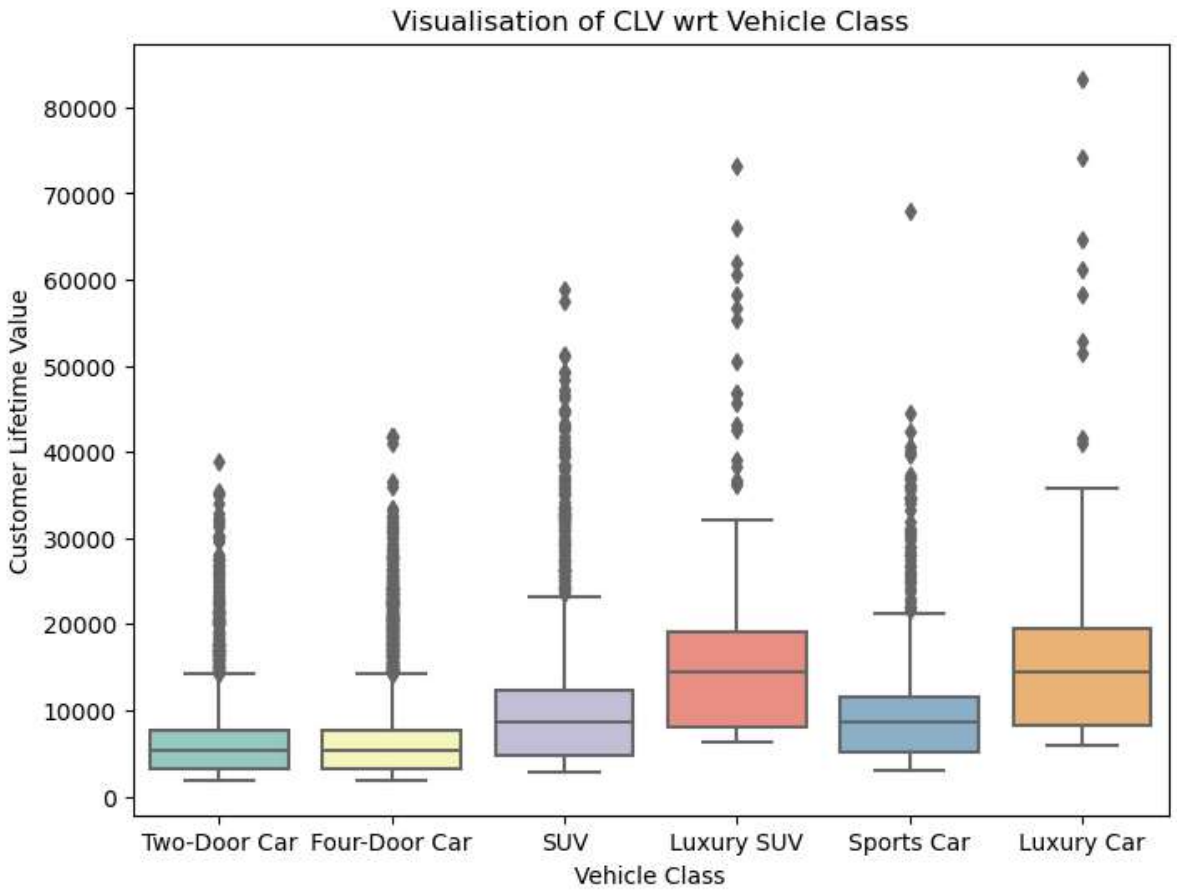


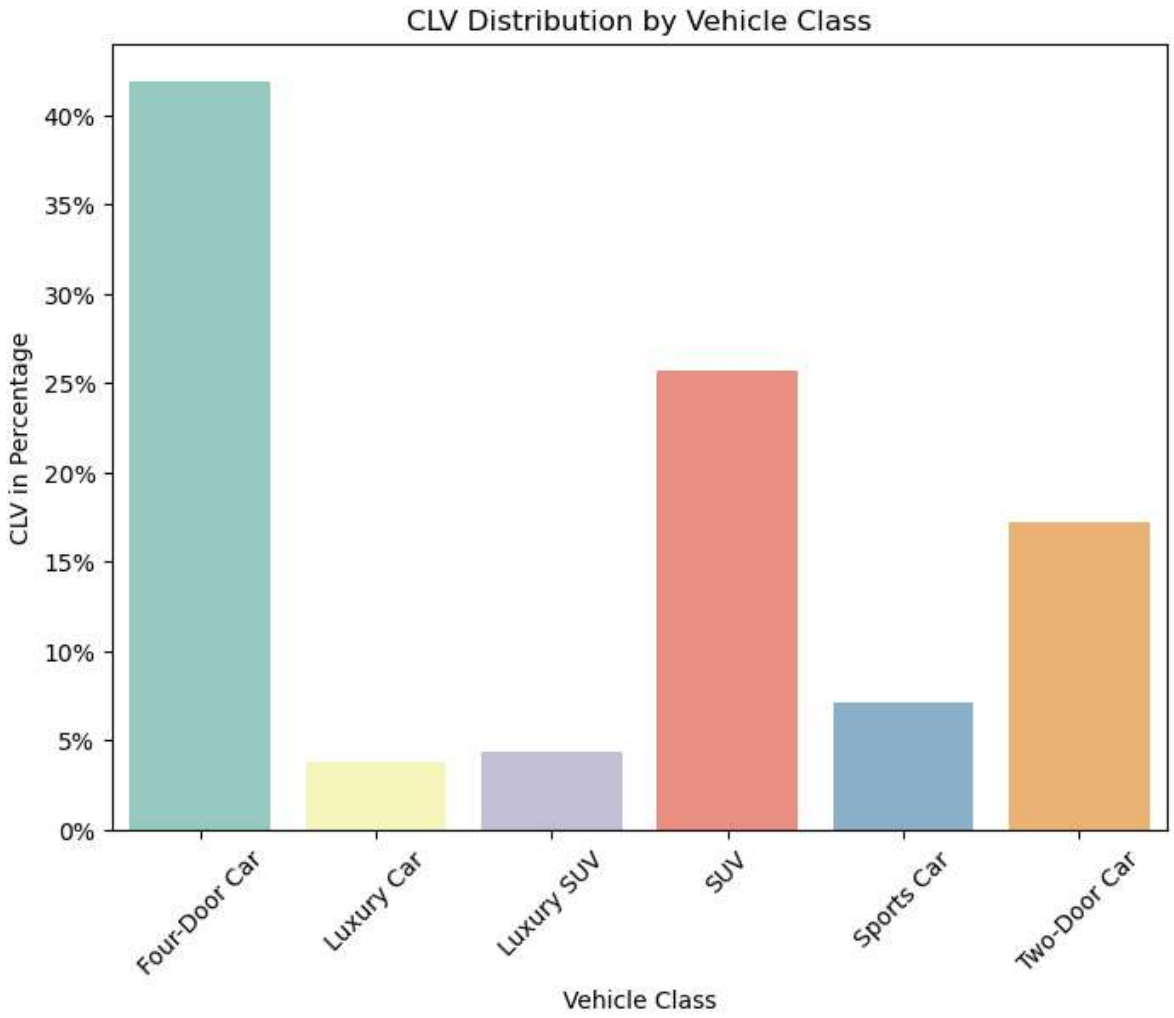
Call Center is not performing well comparerd to other channels throughout the country (in terms of high value customers)

Effect of Vehicle Class on Customer Life Time Value (CLV)

In [32]:

```
# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='Vehicle Class', y='Customer Lifetime Value', data=InsuranceData
plt.xlabel('Vehicle Class')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Vehicle Class')
plt.show()

# Bar chart
aggData = InsuranceData.groupby('Vehicle Class')['Customer Lifetime Value'].su

plt.figure(figsize=(8, 6))
sns.barplot(x='Vehicle Class', y=aggData['Customer Lifetime Value'] / aggData[
plt.xlabel('Vehicle Class')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Vehicle Class')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```



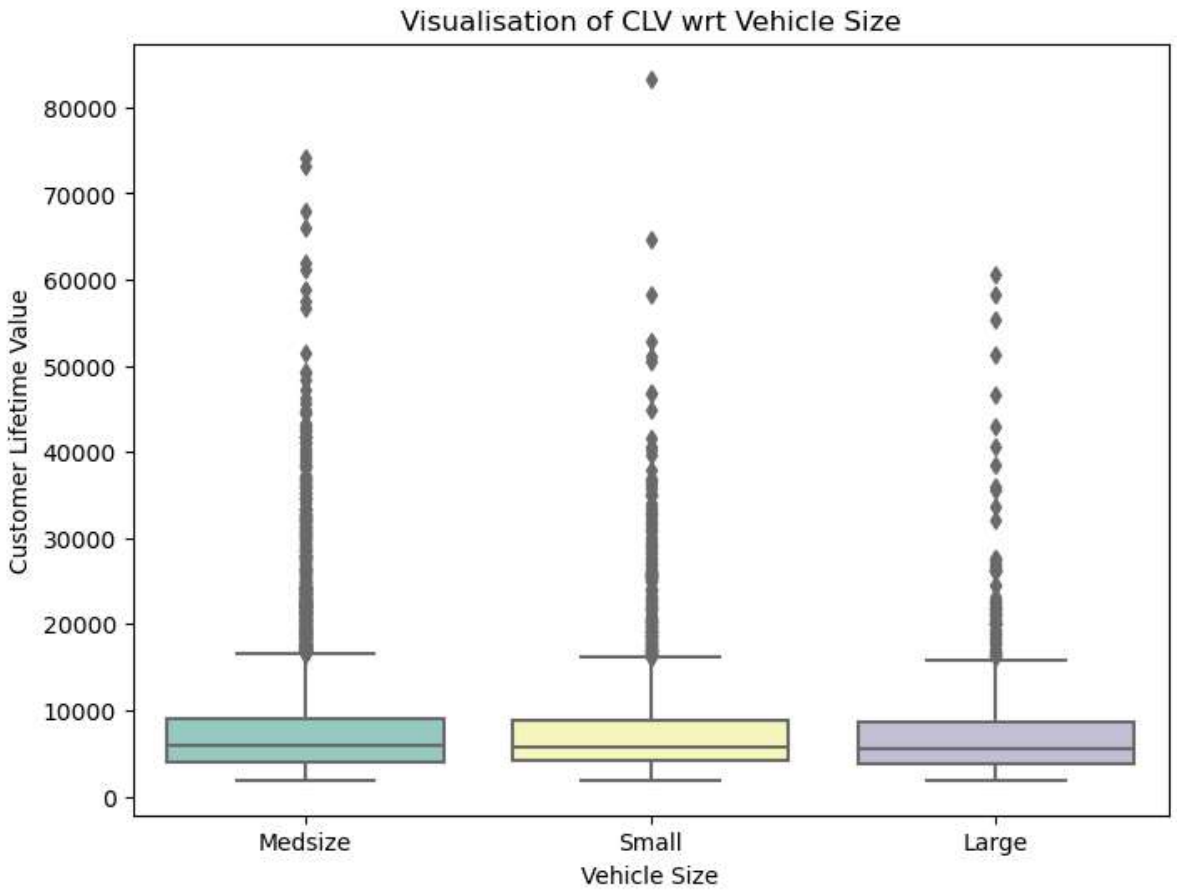


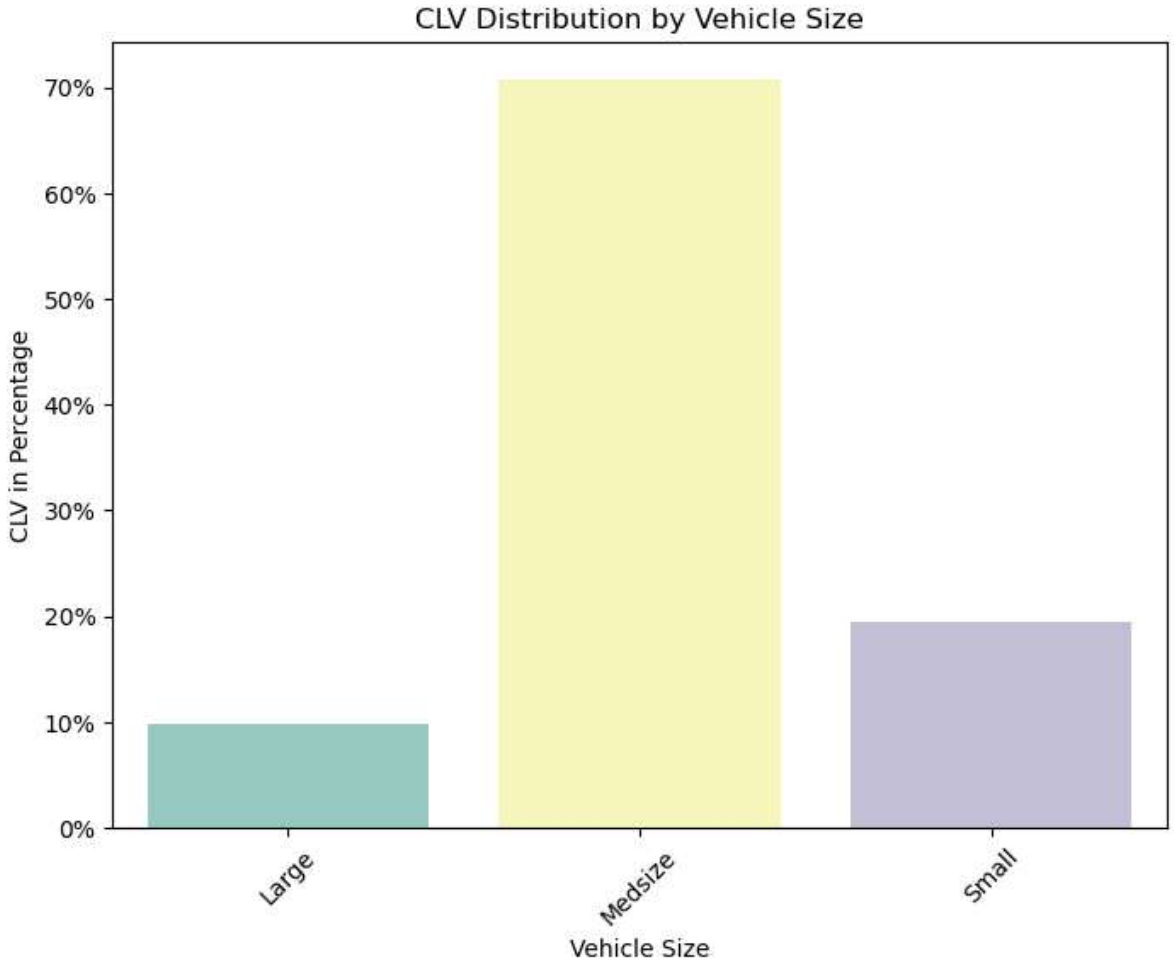
Customers having Four-Door car and SUV are more valuable.

Effect of Vehicle Size on Customer Life Time Value (CLV)

In [33]:

```
# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='Vehicle Size', y='Customer Lifetime Value', data=InsuranceData,
plt.xlabel('Vehicle Size')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Vehicle Size')
plt.show()

# Bar chart
aggData = InsuranceData.groupby('Vehicle Size')['Customer Lifetime Value'].sum

plt.figure(figsize=(8, 6))
sns.barplot(x='Vehicle Size', y=aggData['Customer Lifetime Value'] / aggData['
plt.xlabel('Vehicle Size')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Vehicle Size')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```



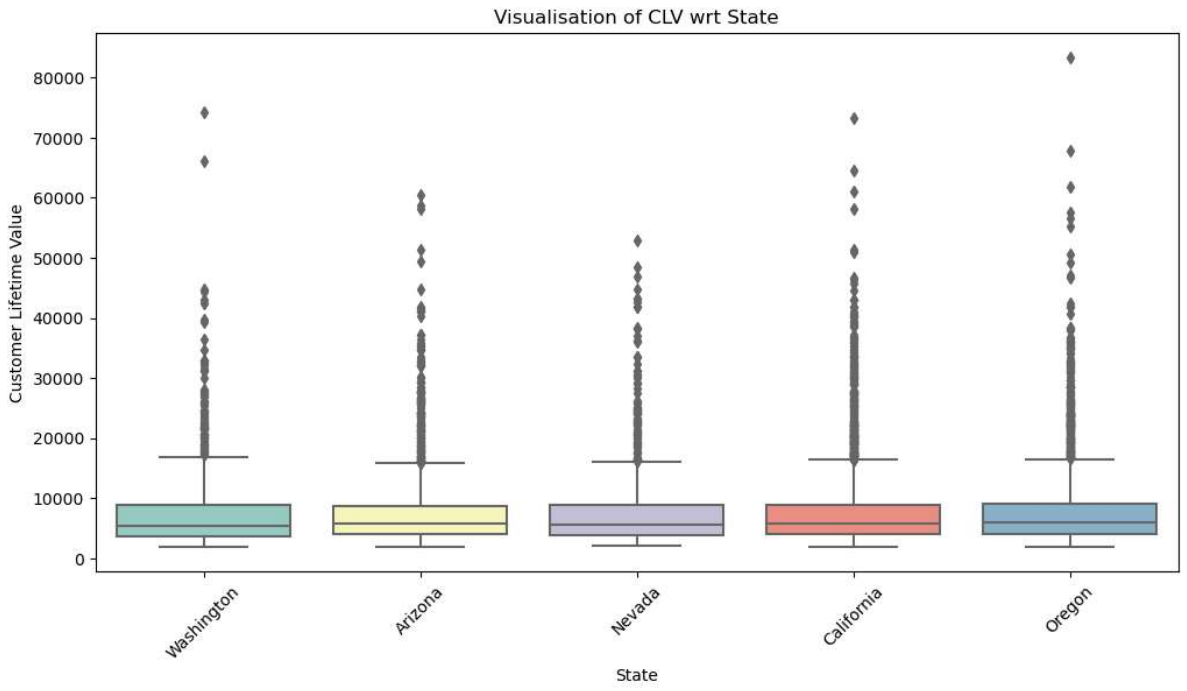


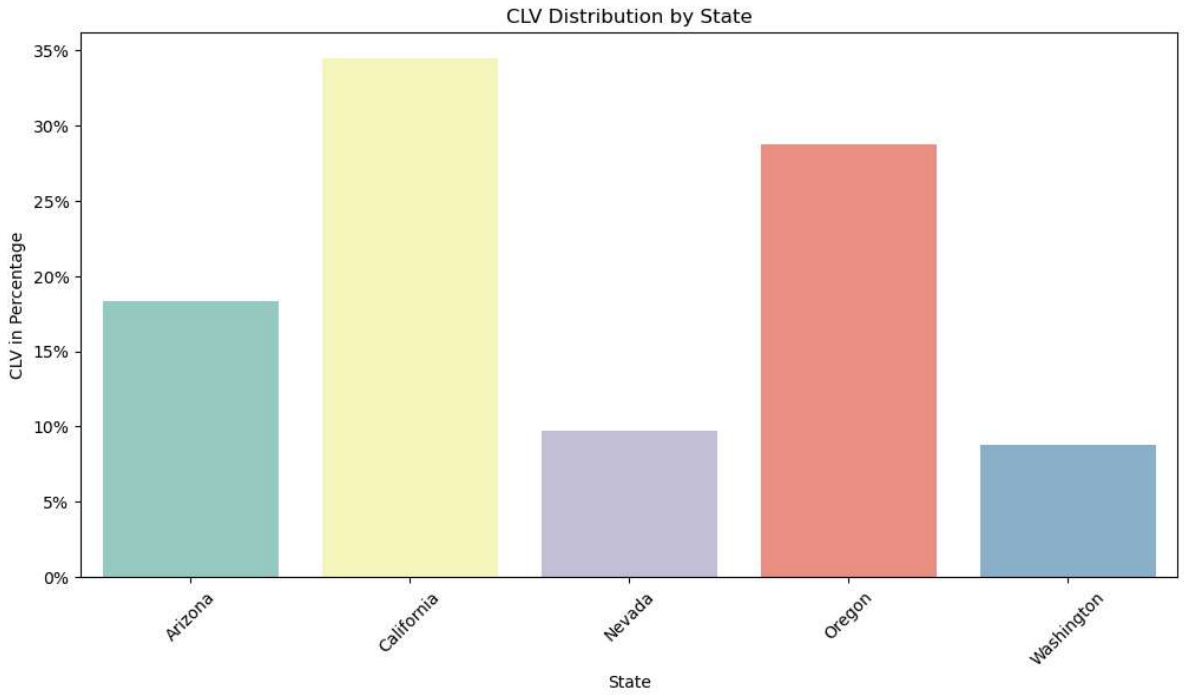
Customers having Mid Size vehicals are adding more value to Insurance company.

Effect of States on Customer Life Time Value (CLV)

In [34]:

```
# Boxplot
plt.figure(figsize=(12, 6))
sns.boxplot(x='State', y='Customer Lifetime Value', data=InsuranceData, palett
plt.xlabel('State')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt State')
plt.xticks(rotation=45)
plt.show()

# Bar chart
aggData = InsuranceData.groupby('State')['Customer Lifetime Value'].sum().rese

plt.figure(figsize=(12, 6))
sns.barplot(x='State', y=aggData['Customer Lifetime Value'] /
            aggData['Customer Lifetime Value'].sum(), data=aggData, palette='S
plt.xlabel('State')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by State')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```

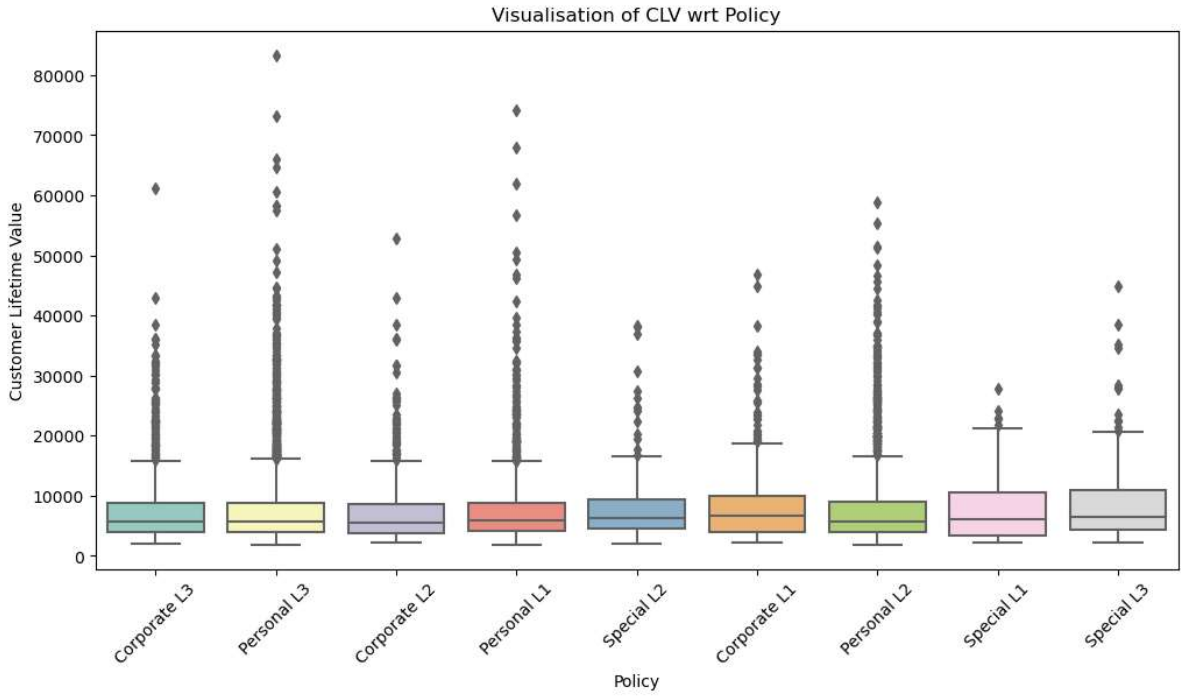




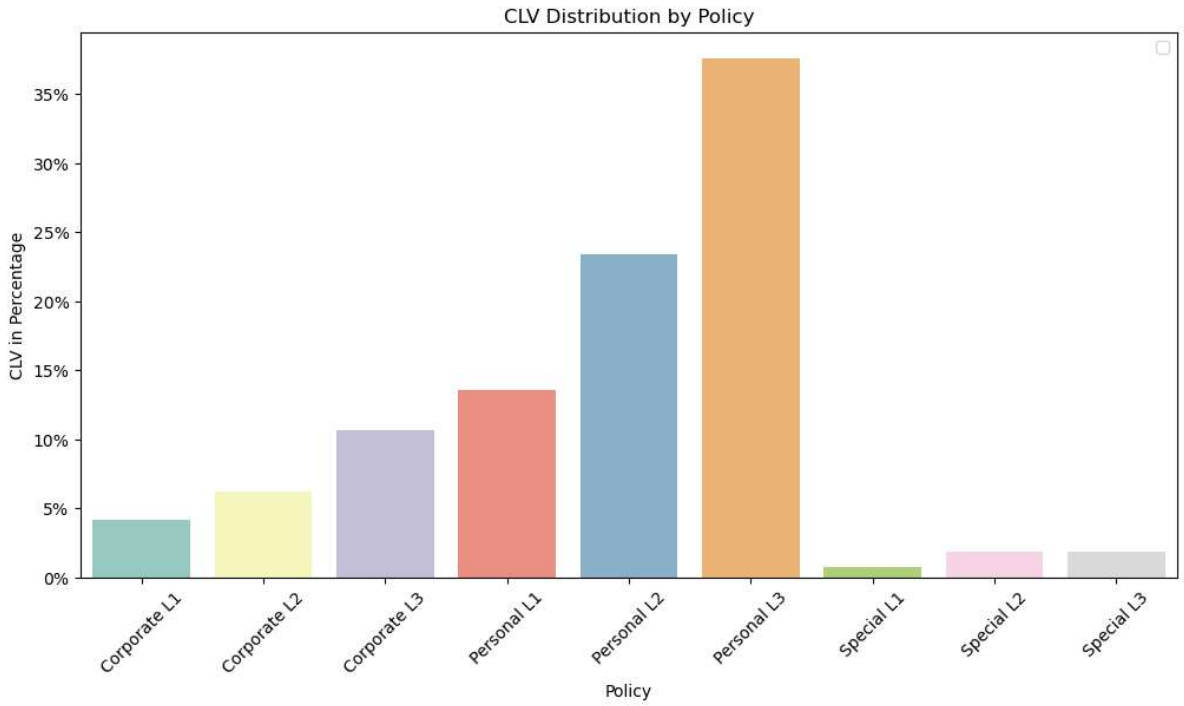
California customers are more valuable.

Effect of Policy on Customer Life Time Value (CLV)

In [35]:
```
# Boxplot
plt.figure(figsize=(12, 6))
sns.boxplot(x='Policy', y='Customer Lifetime Value', data=InsuranceData, palet
plt.xlabel('Policy')
plt.ylabel('Customer Lifetime Value')
plt.title('Visualisation of CLV wrt Policy')
plt.xticks(rotation=45)
plt.show()

# Bar chart
aggData = InsuranceData.groupby('Policy')['Customer Lifetime Value'].sum().res

plt.figure(figsize=(12, 6))
sns.barplot(x='Policy', y=aggData['Customer Lifetime Value'] /
            aggData['Customer Lifetime Value'].sum(), data=aggData, palette='S
plt.xlabel('Policy')
plt.ylabel('CLV in Percentage')
plt.title('CLV Distribution by Policy')
plt.xticks(rotation=45)
plt.legend()
plt.gca().yaxis.set_major_formatter('{:.0%}'.format)
plt.show()
```




```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argumen
t.
```



Personal L3 Policy is adding more value to company.

HERE ENDS OUR EDA SECTION NOW WE MOVE TOWARDS MACHINE LEARNING.

In [12]:
```
InsuranceData_cat=InsuranceData.select_dtypes(include="object").columns
InsuranceData_cat
```

Out[12]:
```
Index(['State', 'Response', 'Coverage', 'Education', 'Effective To Date',
       'EmploymentStatus', 'Gender', 'Location Code', 'Marital Status',
       'Policy Type', 'Policy', 'Renew Offer Type', 'Sales Channel',
       'Vehicle Class', 'Vehicle Size'],
      dtype='object')
```

In [13]:
```
# Convert the 'date' column to datetime
InsuranceData['Effective To Date'] = pd.to_datetime(InsuranceData['Effective T
```

In [14]:
```
InsuranceData_cat=InsuranceData.select_dtypes(include="object").columns
InsuranceData_cat
```

Out[14]:
```
Index(['State', 'Response', 'Coverage', 'Education', 'EmploymentStatus',
       'Gender', 'Location Code', 'Marital Status', 'Policy Type', 'Policy',
       'Renew Offer Type', 'Sales Channel', 'Vehicle Class', 'Vehicle Size'],
      dtype='object')
```

In [15]:
```
continous_var_df =InsuranceData.select_dtypes(include=['int64','float'])
continous_var_df.nunique()
```

Out[15]:
```
Customer Lifetime Value          8041
Income                           5694
Monthly Premium Auto              202
Months Since Last Claim            36
Months Since Policy Inception     100
Number of Open Complaints           6
Number of Policies                  9
Total Claim Amount               5106
dtype: int64
```

In [16]:
```
continous_var_df.columns
```

Out[16]:
```
Index(['Customer Lifetime Value', 'Income', 'Monthly Premium Auto',
       'Months Since Last Claim', 'Months Since Policy Inception',
       'Number of Open Complaints', 'Number of Policies',
       'Total Claim Amount'],
      dtype='object')
```

In [17]:
```
plt.figure(figsize=(10,6))
sns.heatmap(continous_var_df.corr(), annot = True)
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[17], line 1
----> 1 plt.figure(figsize=(10,6))
      2 sns.heatmap(continous_var_df.corr(), annot = True)
      3 plt.show()

NameError: name 'plt' is not defined
```

In [18]:
```
categorical_df = InsuranceData.select_dtypes(include='object')
cat_df = categorical_df
```

In [19]:
```
cat_df.nunique()
```

Out[19]:
```
State              5
Response           2
Coverage           3
Education          5
EmploymentStatus   5
Gender             2
Location Code      3
Marital Status     3
Policy Type        3
Policy             9
Renew Offer Type   4
Sales Channel      4
Vehicle Class      6
Vehicle Size       3
dtype: int64
```

In [20]:
```
cat_df.columns
```

Out[20]:
```
Index(['State', 'Response', 'Coverage', 'Education', 'EmploymentStatus',
       'Gender', 'Location Code', 'Marital Status', 'Policy Type', 'Policy',
       'Renew Offer Type', 'Sales Channel', 'Vehicle Class', 'Vehicle Size'],
      dtype='object')
```

In [21]:
```
cols = ['State', 'Coverage', 'Education', 'EmploymentStatus', 'Gender',
       'Location Code', 'Marital Status', 'Policy Type', 'Policy',
       'Renew Offer Type', 'Sales Channel', 'Vehicle Class', 'Vehicle Size']
```

In [22]:
```
from sklearn.preprocessing import LabelEncoder
lb = LabelEncoder()
for col in cat_df[cols]:
    cat_df[col] = lb.fit_transform(cat_df[col])
```

In [23]:
```
cat_df.head()
```

Out[23]:

	State	Response	Coverage	Education	EmploymentStatus	Gender	Location Code	Marital Status	Policy Type
0	4	No	0	0	1	0	1	1	0
1	0	No	1	0	4	0	1	2	1
2	2	No	2	0	1	0	1	1	1
3	1	No	0	0	4	1	1	1	0
4	4	No	0	0	1	1	0	2	1

In [24]:
```
cat_df['Response'] = cat_df['Response'].replace({'Yes': 1, 'No': 0})
```

In [25]:
```
continous_var_df.reset_index(drop = True, inplace=True)
cat_df.reset_index(drop = True, inplace=True)
```

In [26]:
```
all_data_df = pd.concat([continous_var_df,cat_df], axis = 1)
```

In [27]:
```
all_data_df.head()
```

Out[27]:

	Customer Lifetime Value	Income	Monthly Premium Auto	Months Since Last Claim	Months Since Policy Inception	Number of Open Complaints	Number of Policies	Total Claim Amount	State
0	2763.519279	56274	69	32	5	0	1	384.811147	4
1	6979.535903	0	94	13	42	0	8	1131.464935	0
2	12887.431650	48767	108	18	38	0	2	566.472247	2
3	7645.861827	0	106	18	65	0	7	529.881344	1
4	2813.692575	43836	73	12	44	0	1	138.130879	4

5 rows × 22 columns

In [28]:
```
# Separate features (X) and target variable (y)
X = all_data_df.drop('Customer Lifetime Value', axis=1)
y = all_data_df['Customer Lifetime Value']
y=np.log(y)
```

In [29]:
```
from sklearn.model_selection import train_test_split, cross_validate
X_train,  X_test, y_train, y_test = train_test_split(X,y, test_size = 0.3, ran
```

In [30]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_train.shape)
```

```
(6393, 21)
(2741, 21)
(6393,)
(6393,)
```

In [35]:
```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, r2_score


# Create the Linear Regression model
lr = LinearRegression()

# Fit the model to the training data
lr.fit(X_train, y_train)

# Predict using the trained model
y_pred = lr.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("LINEAR REGRESSION:")
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)

# Perform cross-validation and print the mean R-squared
cross_val_scores = cross_val_score(lr, X_train, y_train, cv=5)
mean_r2_cv = cross_val_scores.mean()
print("Cross Validation Mean R2:", mean_r2_cv)
```

```
LINEAR REGRESSION:
Mean Squared Error (MSE): 0.33728614120460254
R-squared (R2): 0.24569466674515084
Cross Validation Mean R2: 0.24340909248207887
```

In [34]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, r2_score

rf_regressor = RandomForestRegressor()
rf_regressor.fit(X_train, y_train)
y_pred = rf_regressor.predict(X_test)

# Evaluate the model using mean squared error (MSE) and R-squared (R2)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("RANDOM FOREST REGRESSION:")
print('Mean Squared Error (MSE):', mse)
print('R-squared (R2):', r2)

# Perform cross-validation
cross_val_scores = cross_val_score(rf_regressor, X_train, y_train, cv=5)
print('Cross Validation Mean R2:', cross_val_scores.mean())
```

```
RANDOM FOREST REGRESSION:
Mean Squared Error (MSE): 0.039489933799926394
R-squared (R2): 0.9116848751366398
Cross Validation Mean R2: 0.9059642487037982
```

In [33]:
```
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, r2_score

# Assuming X_train, X_test, y_train, y_test are properly defined

# Create the Decision Tree Regressor model
dt_regressor = DecisionTreeRegressor(random_state=42)

# Fit the model to the training data
dt_regressor.fit(X_train, y_train)

# Predict using the trained model
y_pred = dt_regressor.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("DECISION TREE REGRESSION:")
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)

# Perform cross-validation and print the mean R-squared
cross_val_scores = cross_val_score(dt_regressor, X_train, y_train, cv=5)
mean_r2_cv = cross_val_scores.mean()
print("Cross Validation Mean R2:", mean_r2_cv)
```

```
DECISION TREE REGRESSION:
Mean Squared Error (MSE): 0.06757122758197487
R-squared (R2): 0.8488839856934943
Cross Validation Mean R2: 0.8303082940170468
```

Based on the results, the Random Forest Regression model seems to be performing the best among the three. It has the lowest Mean Squared Error (indicating better prediction accuracy) and the highest R-squared values (indicating a better fit to the data). The Cross Validation Mean R2 values also suggest that the Random Forest model generalizes well across different folds of the data.

In [ ]: