mini-project-2

June 25, 2025

[ ]:
```
import seaborn as sns
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

##1. Load Dataset

[ ]:
```
data = pd.read_csv('Vehicle_Insurance.csv')
data
```

[ ]:

	id	Gender	Age	Driving_License	Region_Code	Previously_Insured	\
0	1	Male	44	1	28.0	0	
1	2	Male	76	1	3.0	0	
2	3	Male	47	1	28.0	0	
3	4	Male	21	1	11.0	1	
4	5	Female	29	1	41.0	1	
…	…	…	…	…	…	…	
381104	381105	Male	74	1	26.0	1	
381105	381106	Male	30	1	37.0	1	
381106	381107	Male	21	1	30.0	1	
381107	381108	Female	68	1	14.0	0	
381108	381109	Male	46	1	29.0	0	

	Vehicle_Age	Vehicle_Damage	Annual_Premium	Policy_Sales_Channel	\
0	> 2 Years	Yes	40454.0	26.0	
1	1-2 Year	No	33536.0	26.0	
2	> 2 Years	Yes	38294.0	26.0	
3	< 1 Year	No	28619.0	152.0	
4	< 1 Year	No	27496.0	152.0	
…	…	…	…	…	
381104	1-2 Year	No	30170.0	26.0	
381105	< 1 Year	No	40016.0	152.0	
381106	< 1 Year	No	35118.0	160.0	
381107	> 2 Years	Yes	44617.0	124.0	
381108	1-2 Year	No	41777.0	26.0	

	Vintage	Response
0	217	1

```
1           183         0
2            27         1
3           203         0
4            39         0
…          …       …
381104       88         0
381105      131         0
381106      161         0
381107       74         0
381108      237         0

[381109 rows x 12 columns]
```

##2. Data Loading and Inspection

[ ]:
```
data.head()
```

[ ]:
```
   id  Gender  Age  Driving_License  Region_Code  Previously_Insured  \
0   1    Male   44                1         28.0                   0
1   2    Male   76                1          3.0                   0
2   3    Male   47                1         28.0                   0
3   4    Male   21                1         11.0                   1
4   5  Female   29                1         41.0                   1

  Vehicle_Age Vehicle_Damage  Annual_Premium  Policy_Sales_Channel  Vintage  \
0   > 2 Years            Yes         40454.0                  26.0      217
1    1-2 Year             No         33536.0                  26.0      183
2   > 2 Years            Yes         38294.0                  26.0       27
3    < 1 Year             No         28619.0                 152.0      203
4    < 1 Year             No         27496.0                 152.0       39

   Response
0         1
1         0
2         1
3         0
4         0
```

[ ]:
```
data.tail()
```

[ ]:
```
            id  Gender  Age  Driving_License  Region_Code  Previously_Insured  \
381104  381105    Male   74                1         26.0                   1
381105  381106    Male   30                1         37.0                   1
381106  381107    Male   21                1         30.0                   1
381107  381108  Female   68                1         14.0                   0
381108  381109    Male   46                1         29.0                   0
```

```
       Vehicle_Age Vehicle_Damage  Annual_Premium  Policy_Sales_Channel  \
381104    1-2 Year             No         30170.0                  26.0
381105    < 1 Year             No         40016.0                 152.0
381106    < 1 Year             No         35118.0                 160.0
381107   > 2 Years            Yes         44617.0                 124.0
381108    1-2 Year             No         41777.0                  26.0

        Vintage  Response
381104       88         0
381105      131         0
381106      161         0
381107       74         0
381108      237         0
```

[ ]:
```
data.describe()
```

[ ]:
```
                  id            Age  Driving_License    Region_Code  \
count  381109.000000  381109.000000    381109.000000  381109.000000
mean   190555.000000      38.822584         0.997869      26.388807
std    110016.836208      15.511611         0.046110      13.229888
min         1.000000      20.000000         0.000000       0.000000
25%     95278.000000      25.000000         1.000000      15.000000
50%    190555.000000      36.000000         1.000000      28.000000
75%    285832.000000      49.000000         1.000000      35.000000
max    381109.000000      85.000000         1.000000      52.000000

       Previously_Insured  Annual_Premium  Policy_Sales_Channel  \
count       381109.000000   381109.000000         381109.000000
mean             0.458210    30564.389581            112.034295
std              0.498251    17213.155057             54.203995
min              0.000000     2630.000000              1.000000
25%              0.000000    24405.000000             29.000000
50%              0.000000    31669.000000            133.000000
75%              1.000000    39400.000000            152.000000
max              1.000000   540165.000000            163.000000

             Vintage       Response
count  381109.000000  381109.000000
mean      154.347397       0.122563
std        83.671304       0.327936
min        10.000000       0.000000
25%        82.000000       0.000000
50%       154.000000       0.000000
75%       227.000000       0.000000
max       299.000000       1.000000
```

[ ]:
```
data.isnull().sum()
```

[ ]:
```
id                    0
Gender                0
Age                   0
Driving_License       0
Region_Code           0
Previously_Insured    0
Vehicle_Age           0
Vehicle_Damage        0
Annual_Premium        0
Policy_Sales_Channel  0
Vintage               0
Response              0
dtype: int64
```

[ ]:
```
data.dtypes
```

[ ]:
```
id                      int64
Gender                 object
Age                     int64
Driving_License         int64
Region_Code           float64
Previously_Insured      int64
Vehicle_Age            object
Vehicle_Damage         object
Annual_Premium        float64
Policy_Sales_Channel  float64
Vintage                 int64
Response                int64
dtype: object
```

#####Data cleaning

[ ]:
```
data.duplicated().sum()
```

[ ]:
```
np.int64(0)
```

[ ]:
```
data = data.drop_duplicates()
```

[ ]:
```
Q1 = data['Annual_Premium'].quantile(0.25)
Q3 = data['Annual_Premium'].quantile(0.75)
IQR = Q3 - Q1
```

[ ]:
```
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

[ ]:
```
data = data[(data['Annual_Premium'] >= lower_bound) & (data['Annual_Premium']␣
↪<= upper_bound)]
```

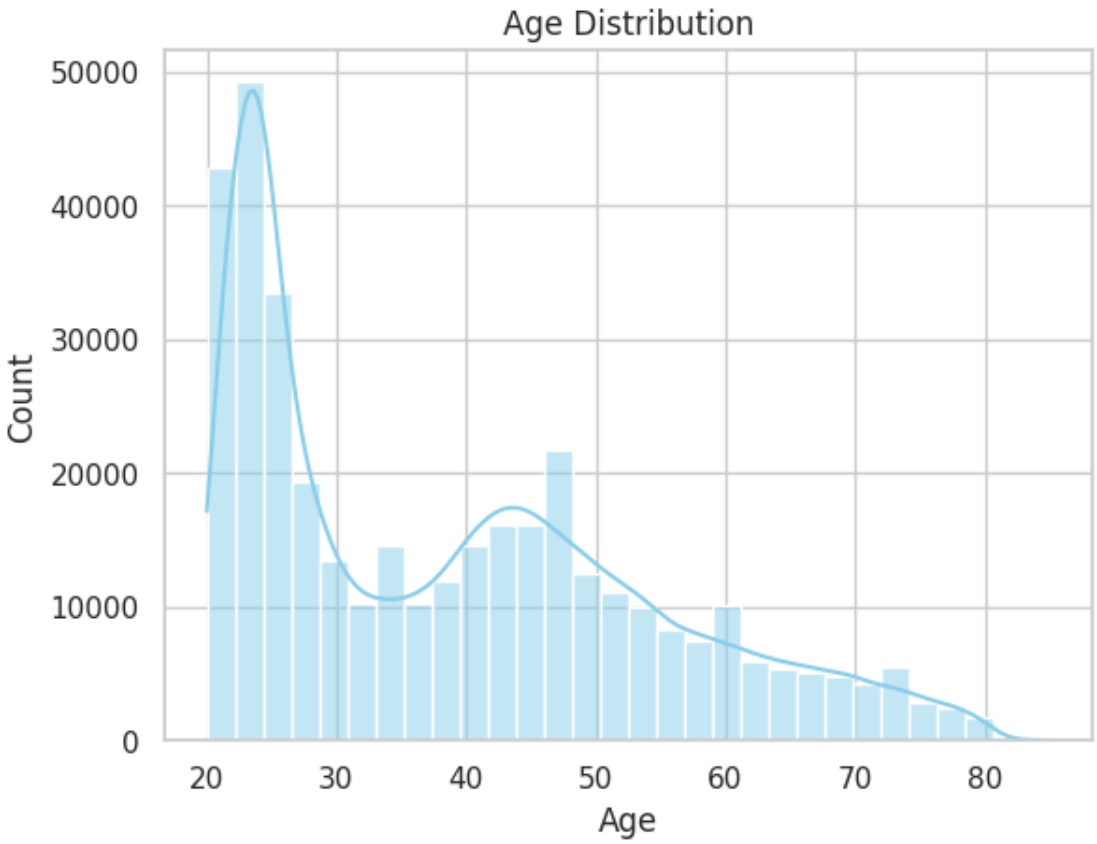
[ ]:
```
data.shape
```

[ ]: (370789, 12)

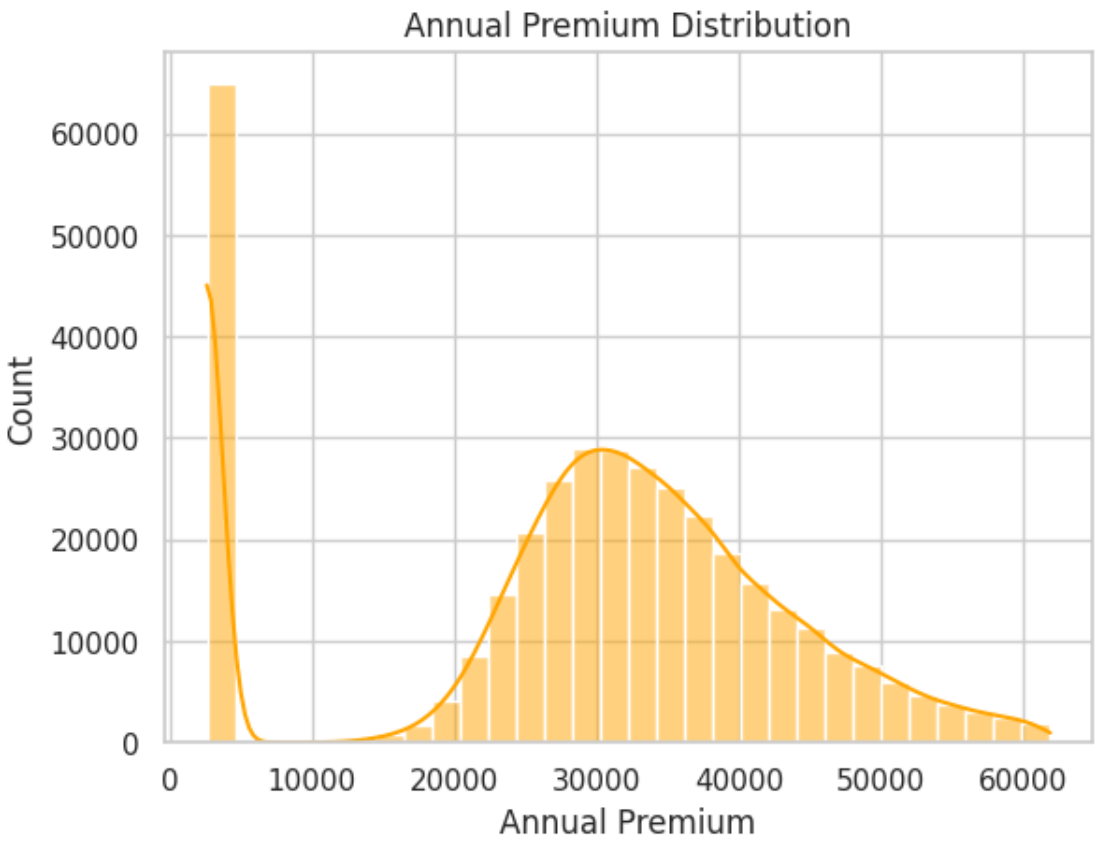
#3. Data Visualisation ###Target Distribution

[ ]:
```
sns.set(style="whitegrid")
```

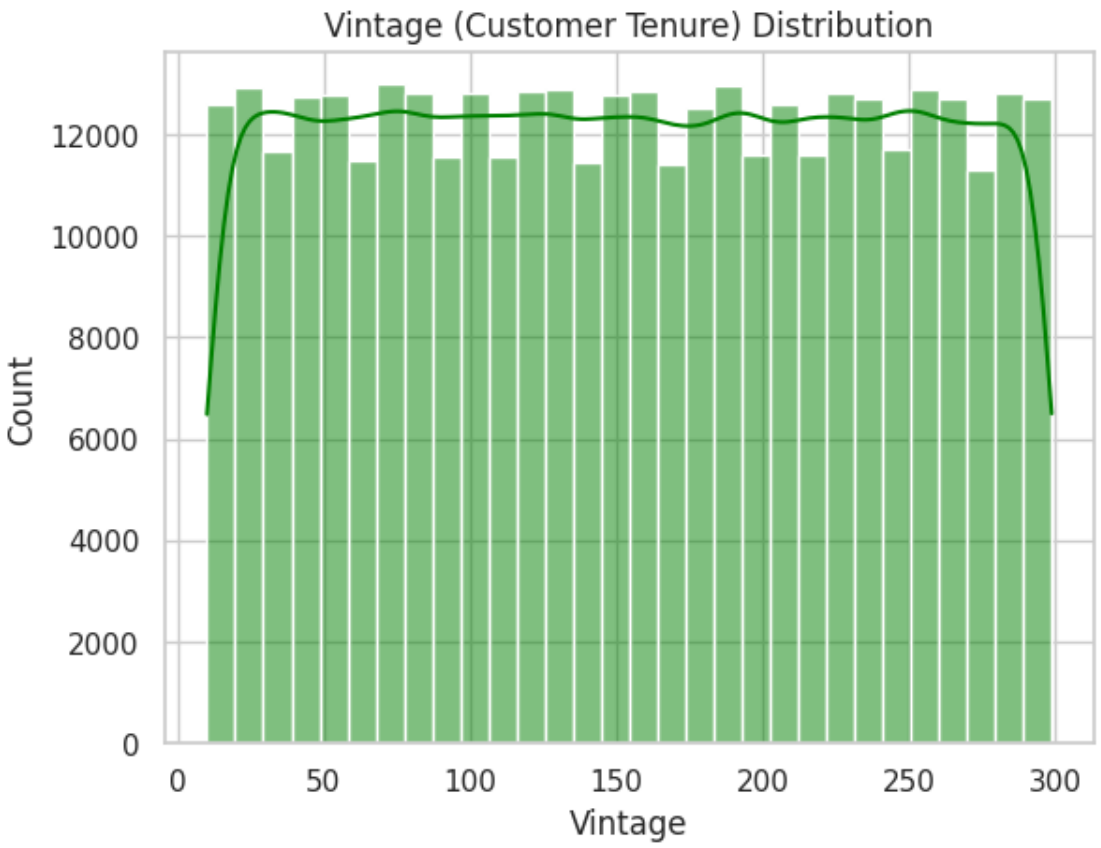
[ ]:
```
sns.histplot(data['Age'], bins=30, kde=True, color='skyblue')
plt.title("Age Distribution")
plt.xlabel("Age")
plt.ylabel("Count")
plt.show()
```



[ ]:
```
sns.histplot(data['Annual_Premium'], bins=30, kde=True, color='orange')
plt.title("Annual Premium Distribution")
plt.xlabel("Annual Premium")
plt.ylabel("Count")
plt.show()
```

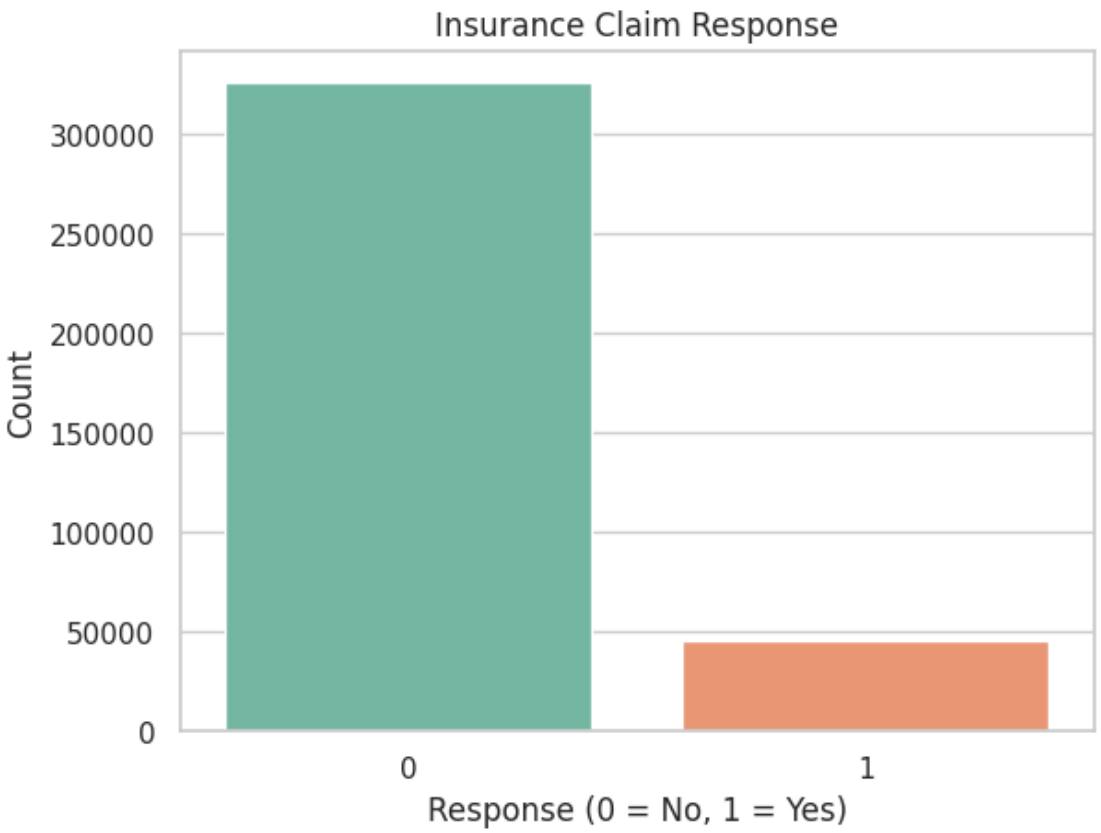


[ ]:
```
sns.histplot(data['Vintage'], bins=30, kde=True, color='green')
plt.title("Vintage (Customer Tenure) Distribution")
plt.xlabel("Vintage")
plt.ylabel("Count")
plt.show()
```



[ ]:
```
sns.countplot(x='Response', data=data, palette='Set2')
plt.title("Insurance Claim Response")
plt.xlabel("Response (0 = No, 1 = Yes)")
plt.ylabel("Count")
plt.show()
```

```
<ipython-input-88-c3471239663a>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.countplot(x='Response', data=data, palette='Set2')
```

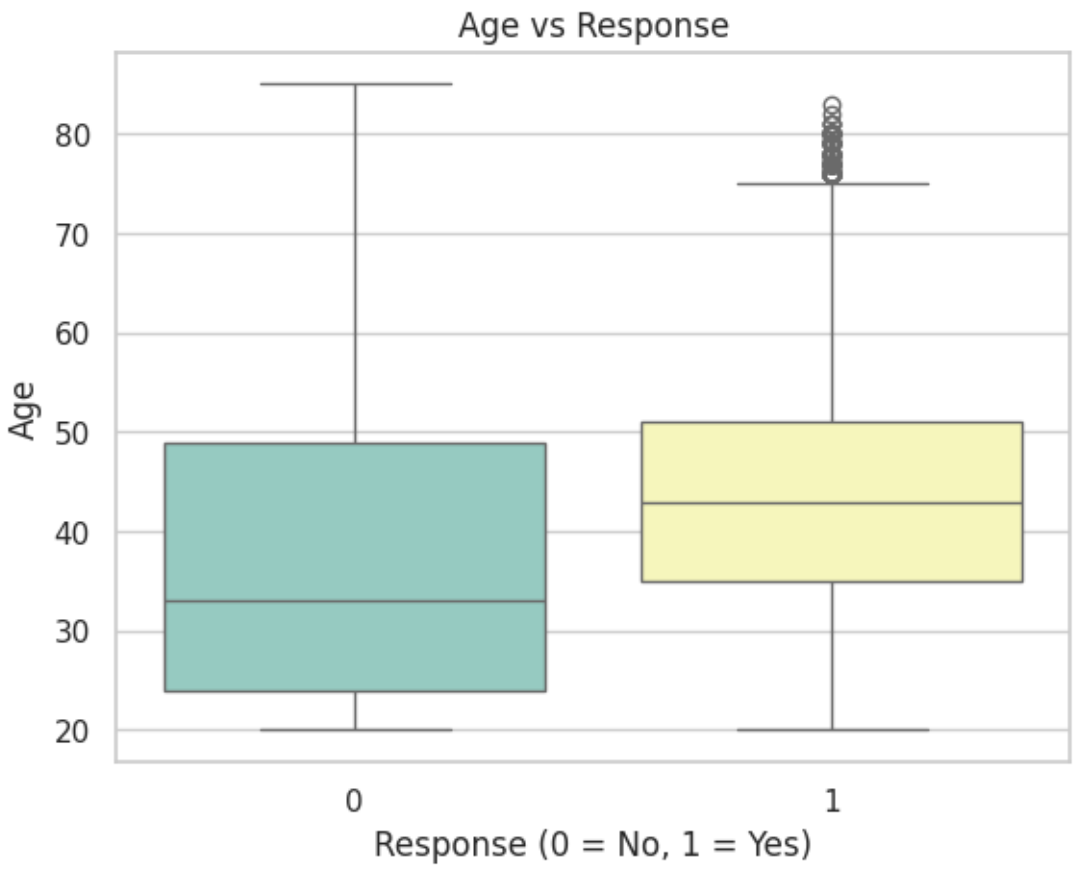


##4. Feature Analysis

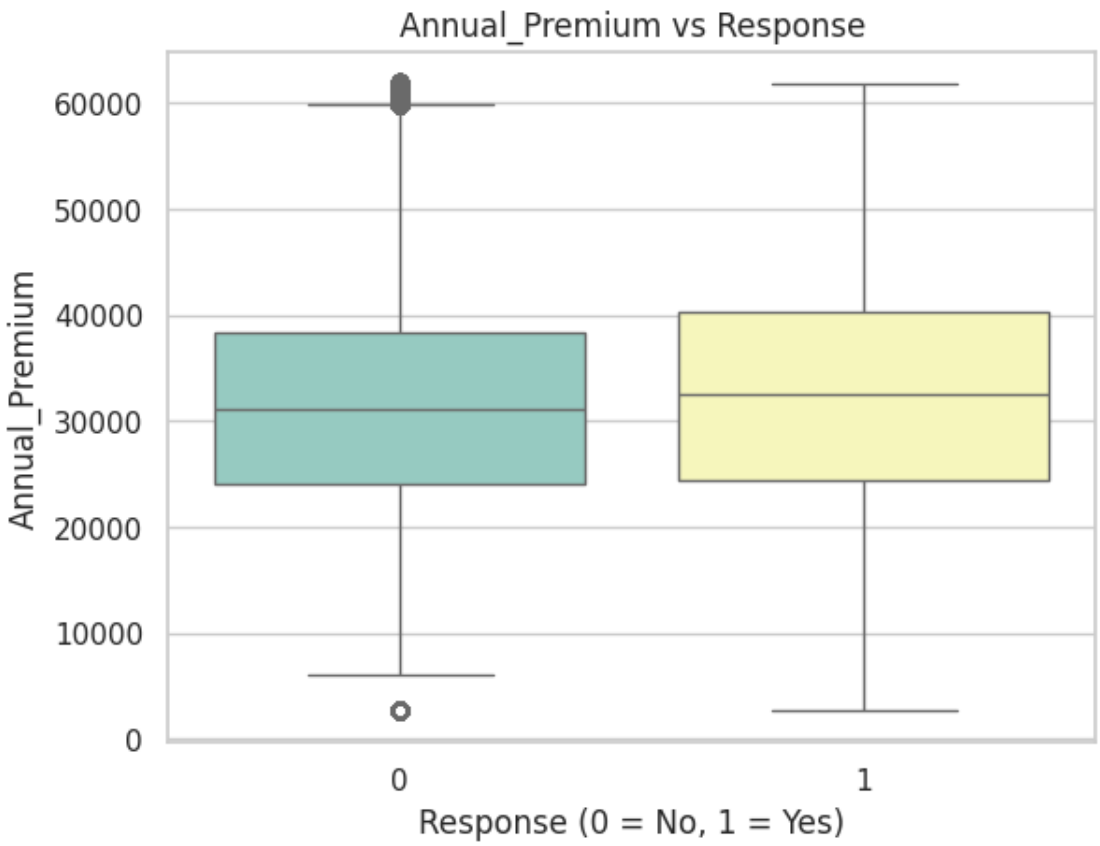
[ ]:
```
num_features = ['Age', 'Annual_Premium', 'Vintage']
```

[ ]:
```
for feature in num_features:
    sns.boxplot(x='Response', y=feature, data=data, palette='Set3')
    plt.title(f'{feature} vs Response')
    plt.xlabel("Response (0 = No, 1 = Yes)")
    plt.ylabel(feature)
    plt.show()
```

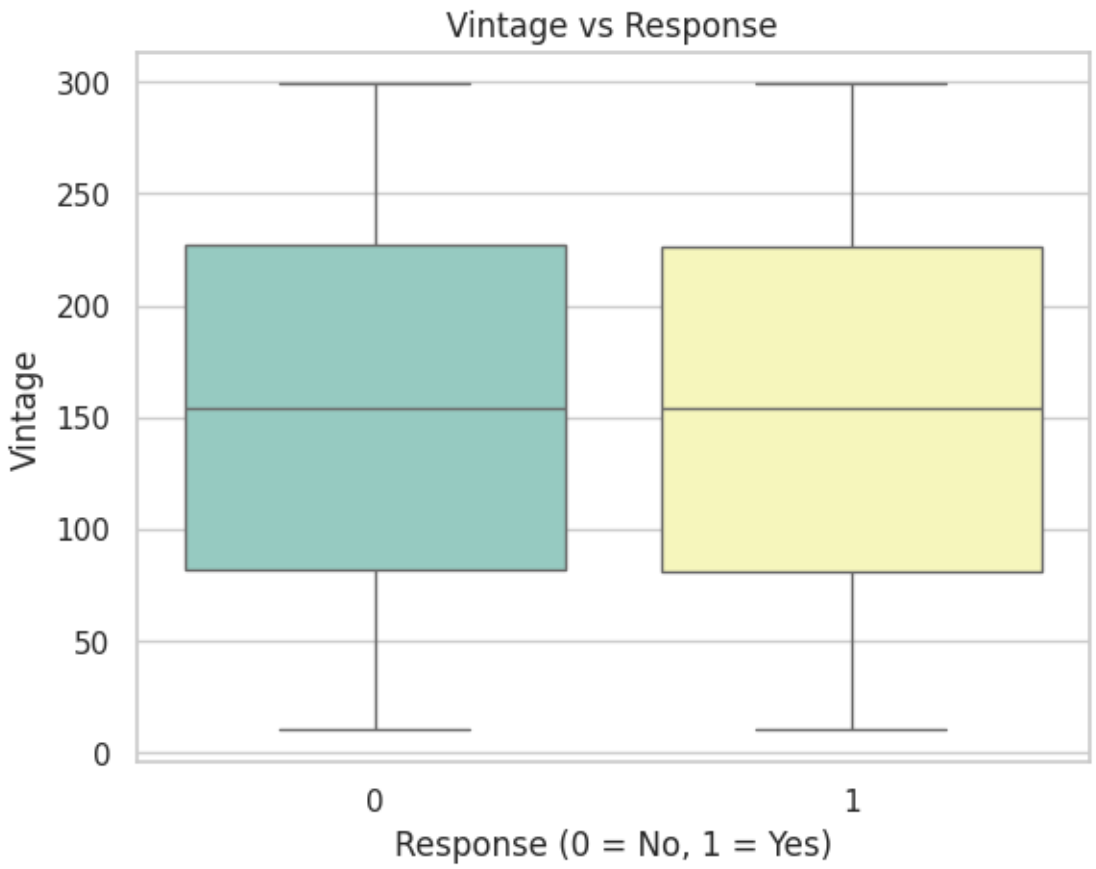
```
<ipython-input-87-2f2f9f1f6f42>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.boxplot(x='Response', y=feature, data=data, palette='Set3')
```




```
<ipython-input-87-2f2f9f1f6f42>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.boxplot(x='Response', y=feature, data=data, palette='Set3')
```

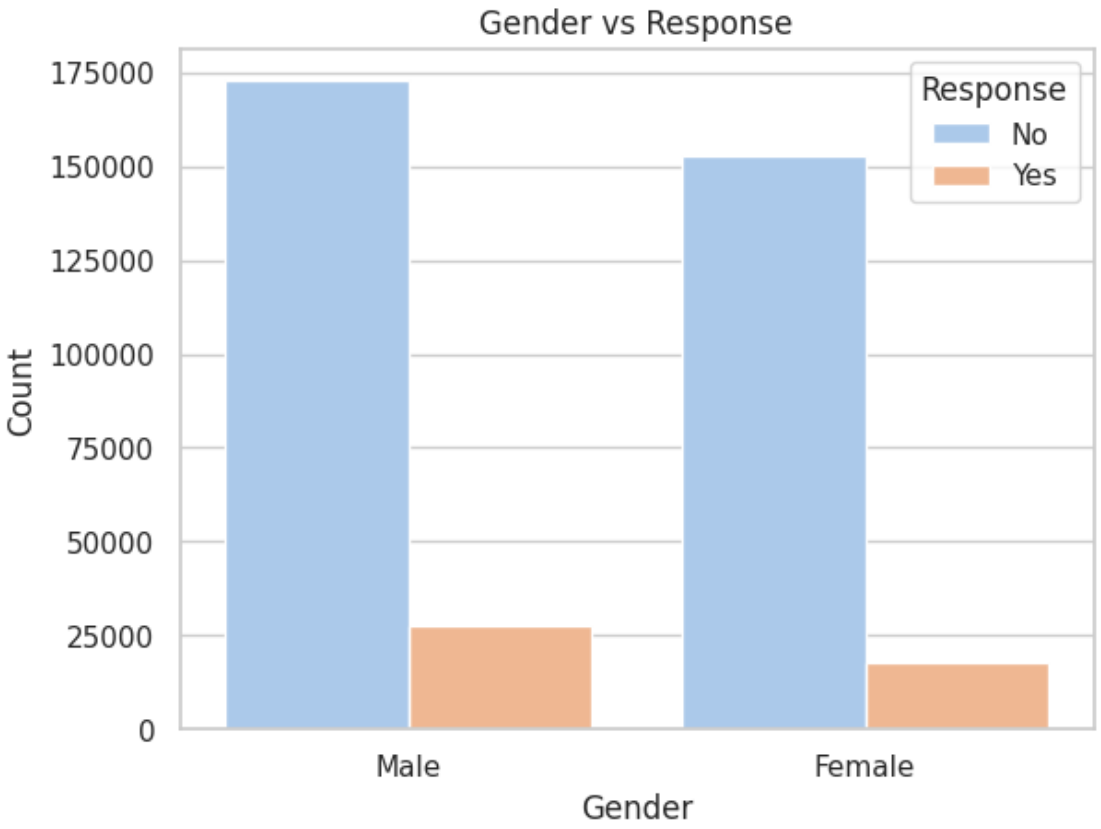


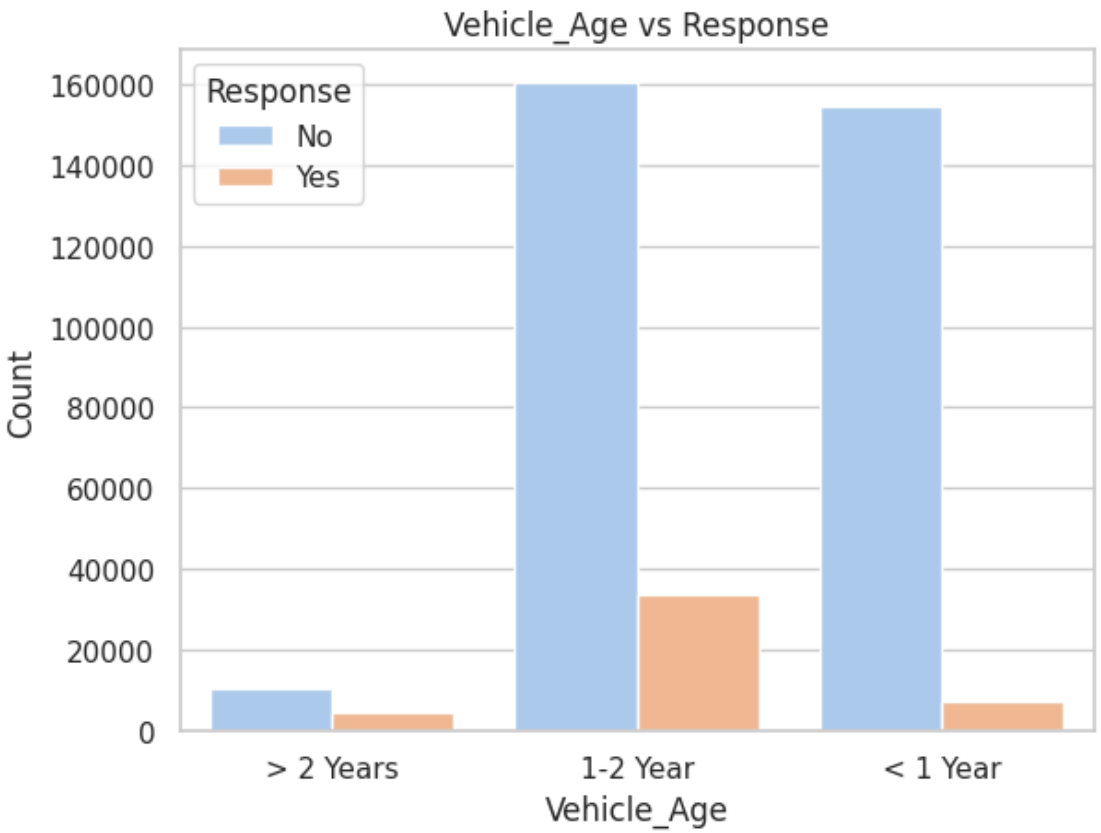

```
<ipython-input-87-2f2f9f1f6f42>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.boxplot(x='Response', y=feature, data=data, palette='Set3')
```

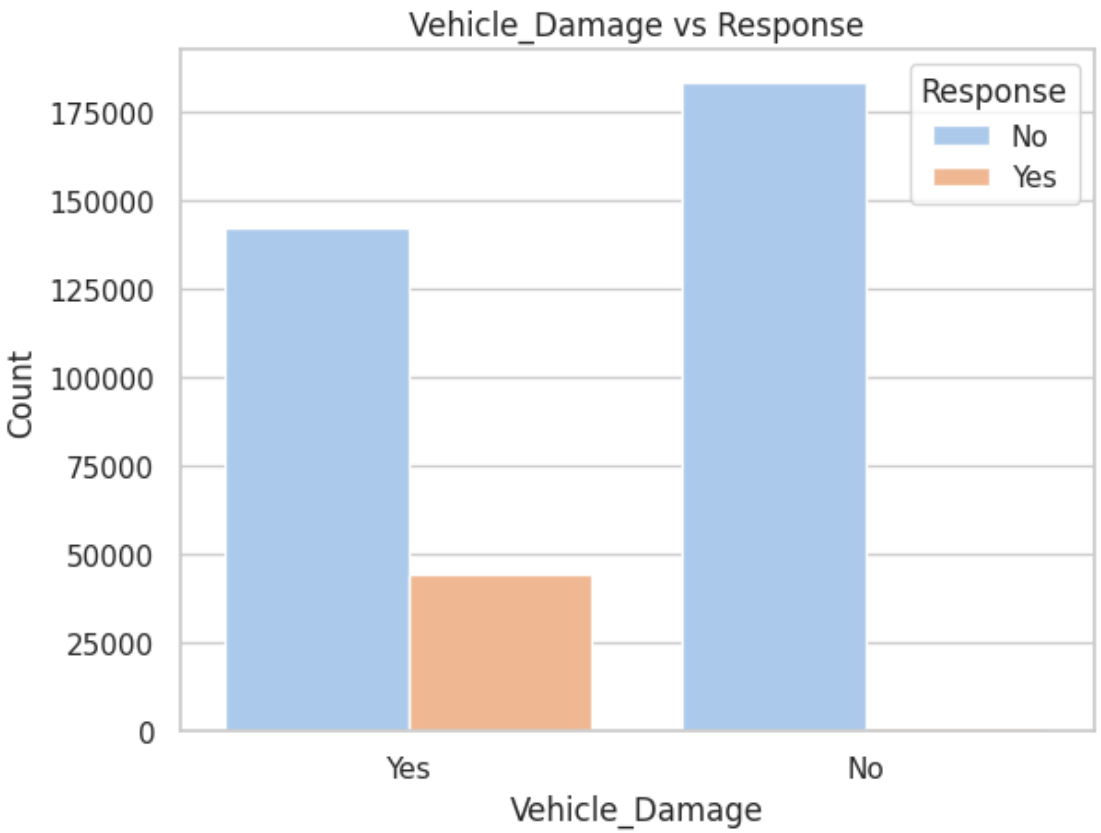


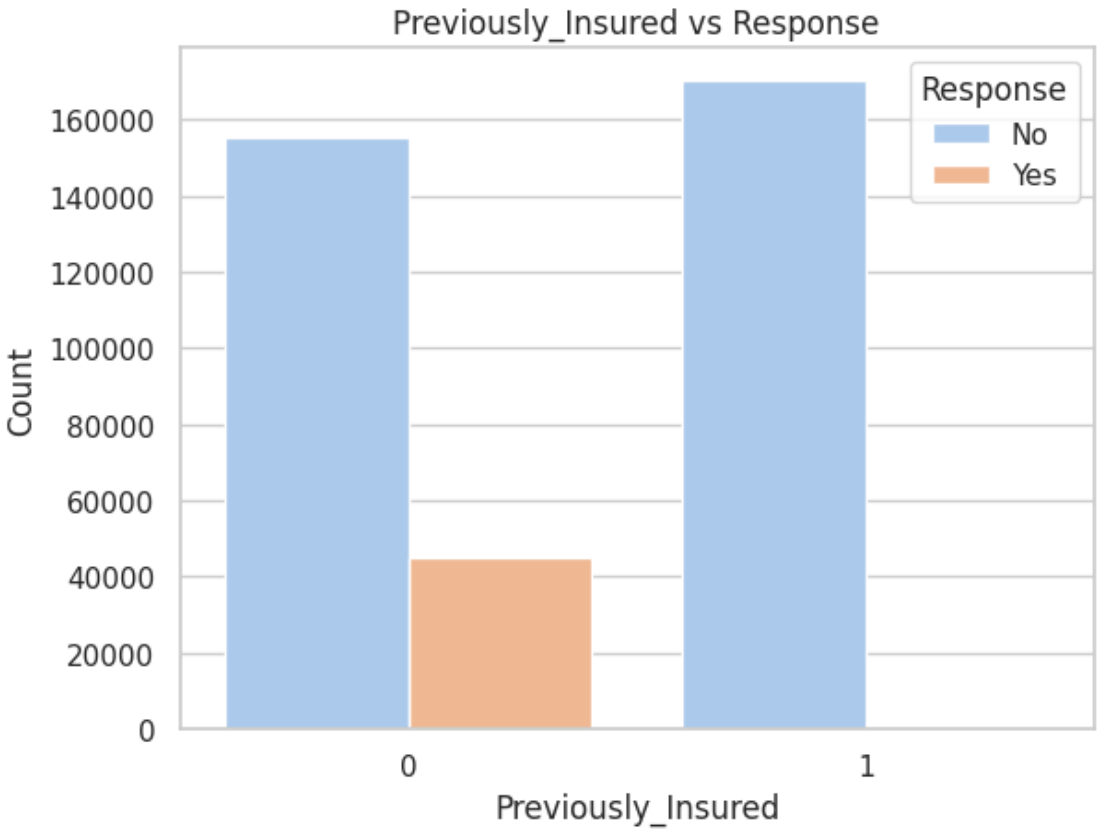
[ ]:
```
cat_features = ['Gender', 'Vehicle_Age', 'Vehicle_Damage', 'Previously_Insured']
```

[ ]:
```
for feature in cat_features:
    sns.countplot(x=feature, hue='Response', data=data, palette='pastel')
    plt.title(f'{feature} vs Response')
    plt.xlabel(feature)
    plt.ylabel("Count")
    plt.legend(title='Response', labels=['No', 'Yes'])
    plt.show()
```


Gender vs Response
Response
No
Yes
175000
150000
125000
100000
75000
50000
25000
0
Count
Male
Female
Gender

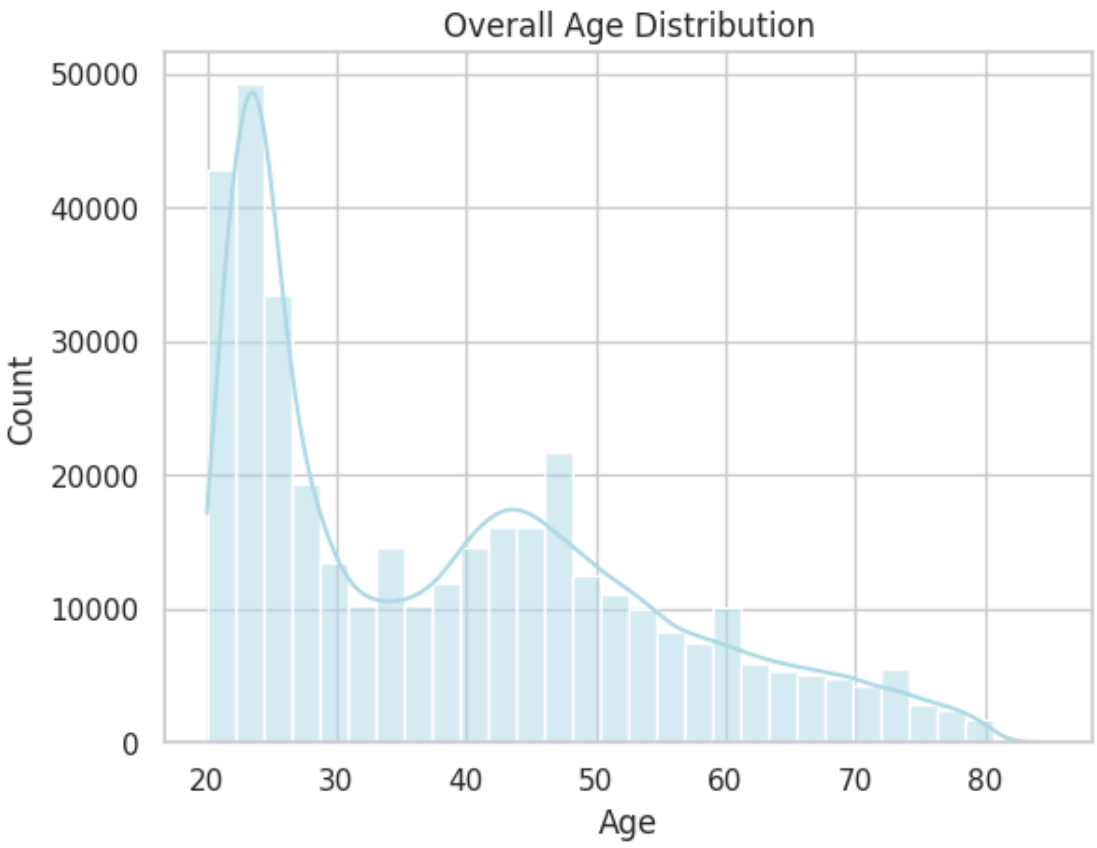

Vehicle_Age vs Response
Response
No
Yes
160000
140000
120000
100000
80000
60000
40000
20000
0
Count
> 2 Years
1-2 Year
< 1 Year
Vehicle_Age


Vehicle_Damage vs Response
Response
No
Yes
175000
150000
125000
100000
75000
50000
25000
0
Count
Yes
No
Vehicle_Damage

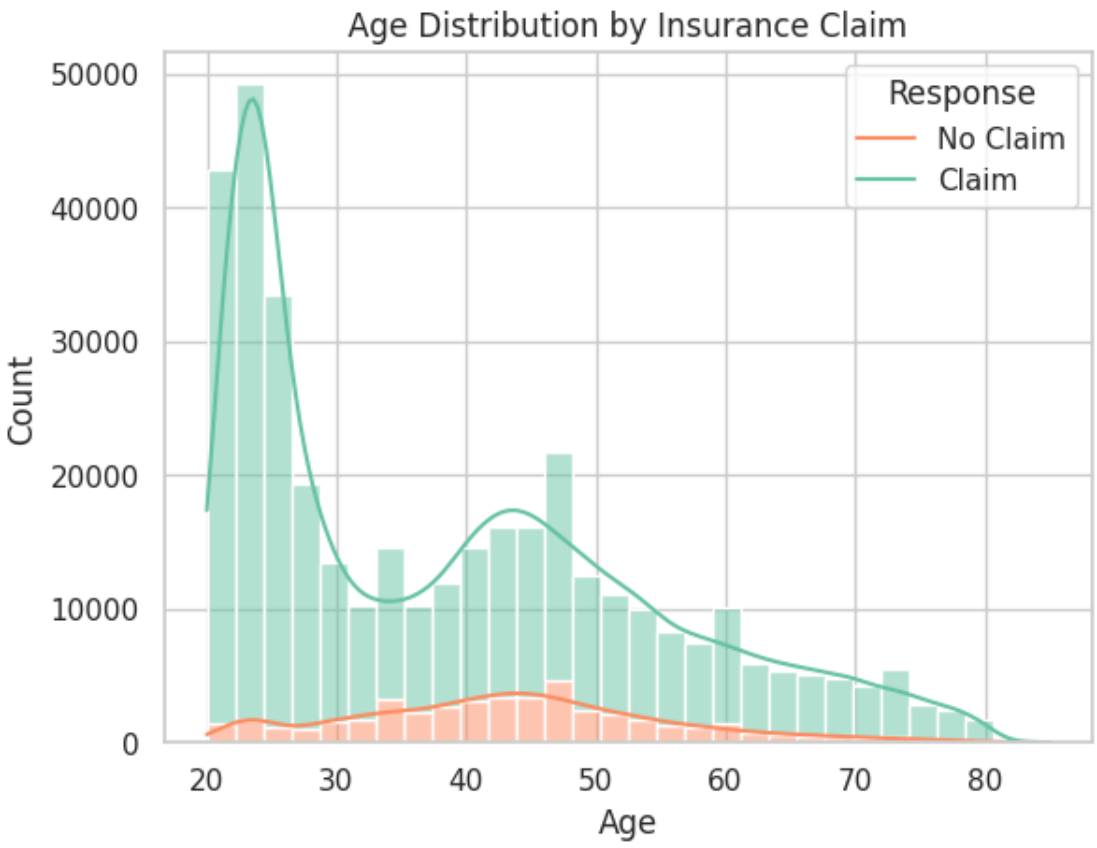


##5. Age Distribution

[ ]:
```
sns.histplot(data['Age'], bins=30, kde=True, color='lightblue')
plt.title('Overall Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



[ ]:
```
sns.histplot(data=data, x='Age', hue='Response', bins=30, kde=True,␣
↪palette='Set2', multiple='stack')
plt.title('Age Distribution by Insurance Claim')
plt.xlabel('Age')
plt.ylabel('Count')
plt.legend(title='Response', labels=['No Claim', 'Claim'])
plt.show()
```



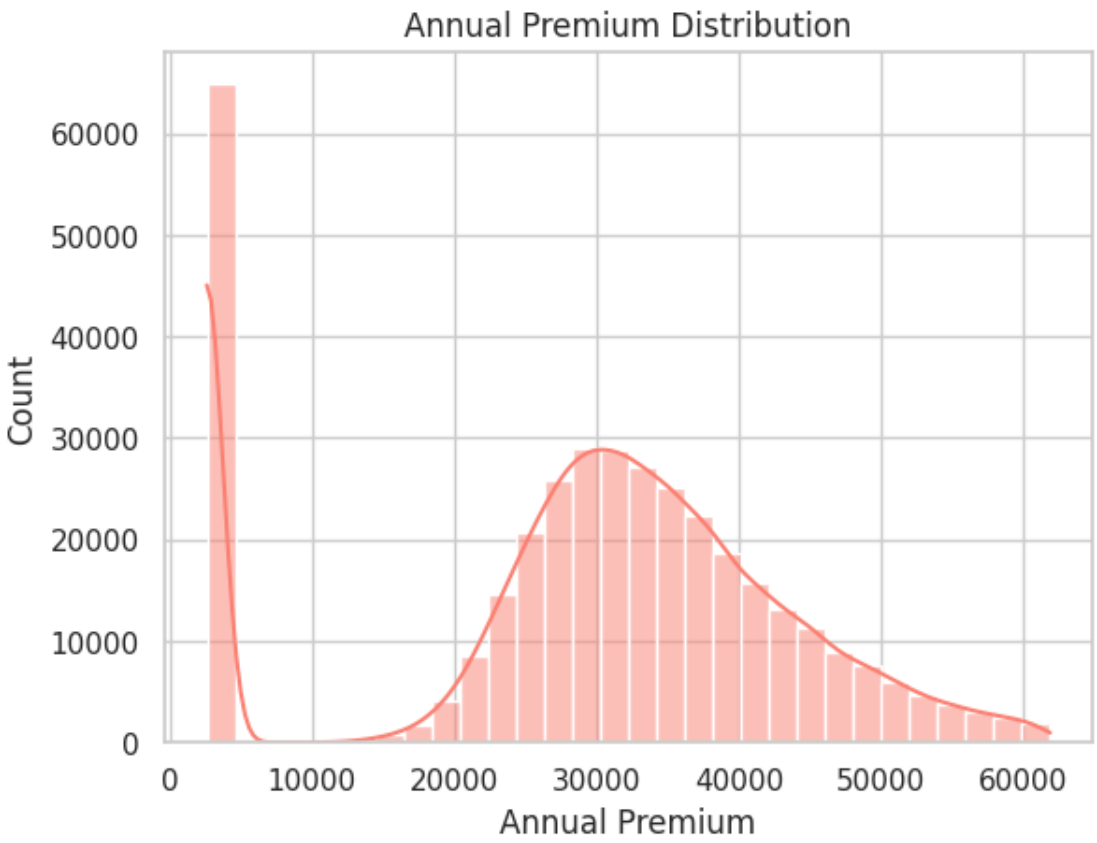
[ ]:
```
mean_age_by_response = data.groupby('Response')['Age'].mean()
```

[ ]:
```
mean_age_by_response
```

[ ]:
```
Response
0    38.032494
1    43.270180
Name: Age, dtype: float64
```

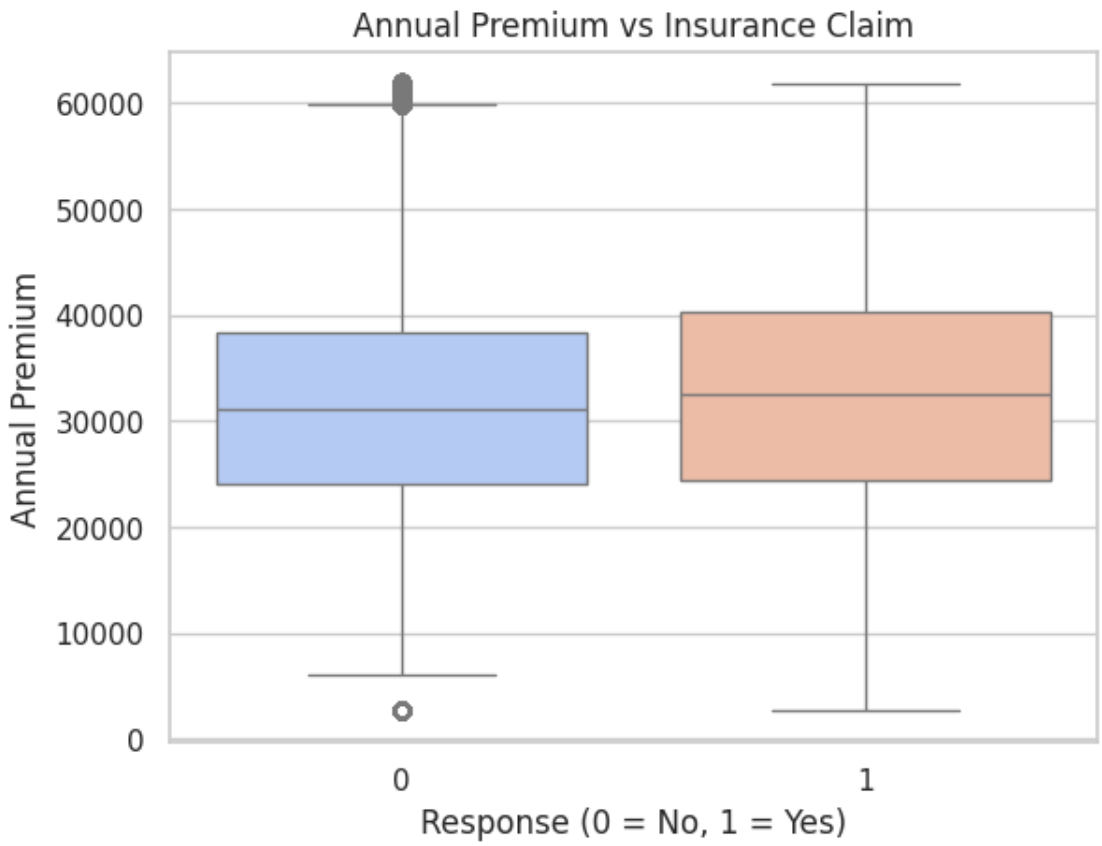
##6. Premium Analysis

[ ]:
```
sns.histplot(data['Annual_Premium'], bins=30, kde=True, color='salmon')
plt.title("Annual Premium Distribution")
plt.xlabel("Annual Premium")
plt.ylabel("Count")
plt.show()
```



[ ]:
```
sns.boxplot(x='Response', y='Annual_Premium', data=data, palette='coolwarm')
plt.title("Annual Premium vs Insurance Claim")
plt.xlabel("Response (0 = No, 1 = Yes)")
plt.ylabel("Annual Premium")
plt.show()
```

```
<ipython-input-82-8ab2b199de66>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.boxplot(x='Response', y='Annual_Premium', data=data, palette='coolwarm')
```



[ ]:
```
mean_premium_by_response = data.groupby('Response')['Annual_Premium'].mean()
```

[ ]:
```
mean_premium_by_response
```

[ ]:
```
Response
0    29162.717548
1    29999.683490
Name: Annual_Premium, dtype: float64
```

##7. Claim Frequencies VS Features

[ ]:
```
claim_rate_by_gender = data.groupby('Gender')['Response'].mean()
```

[ ]:
```
claim_rate_by_gender
```

[ ]:
```
Gender
Female    0.103238
Male      0.137561
Name: Response, dtype: float64
```

[ ]:
```
claim_rate_by_damage = data.groupby('Vehicle_Damage')['Response'].mean()
```

[ ]:
```
claim_rate_by_damage
```

[ ]:
```
Vehicle_Damage
No     0.005249
Yes    0.236856
Name: Response, dtype: float64
```

[ ]:
```
claim_rate_by_insurance = data.groupby('Previously_Insured')['Response'].mean()
```

[ ]:
```
claim_rate_by_insurance
```

[ ]:
```
Previously_Insured
0    0.224612
1    0.000904
Name: Response, dtype: float64
```

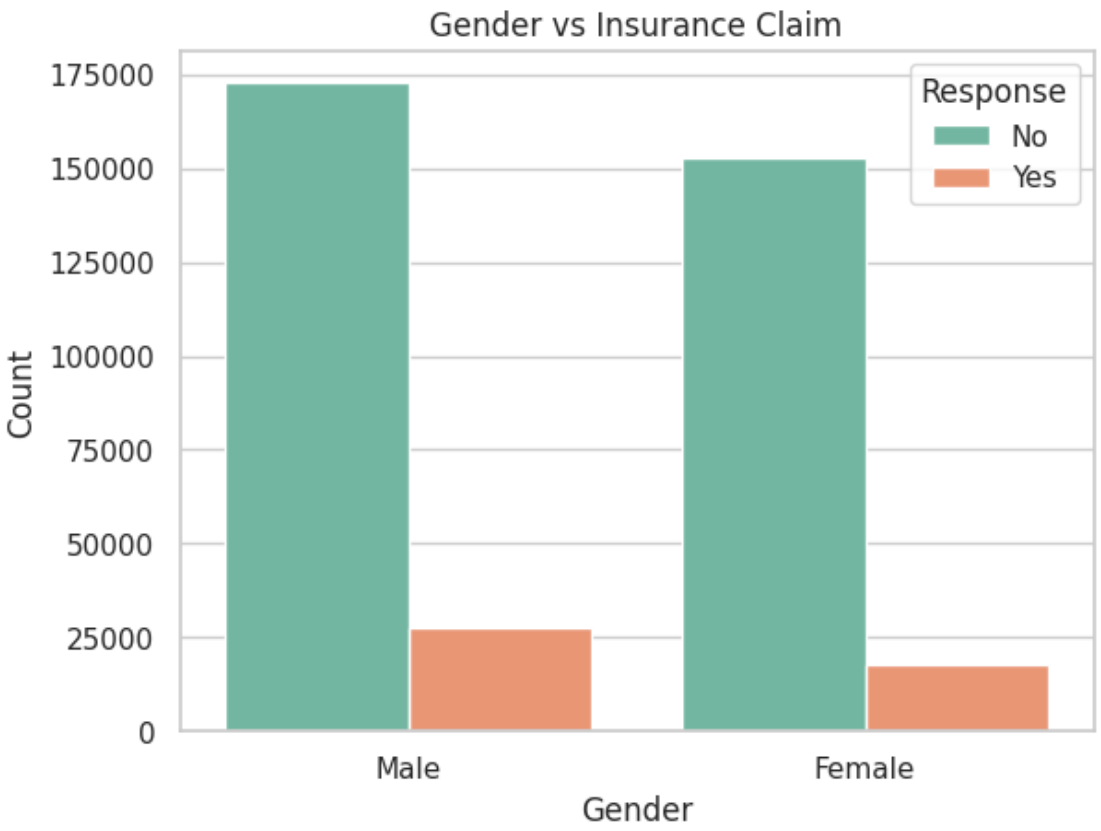
[ ]:
```
claim_rate_by_vehicle_age = data.groupby('Vehicle_Age')['Response'].mean()
```

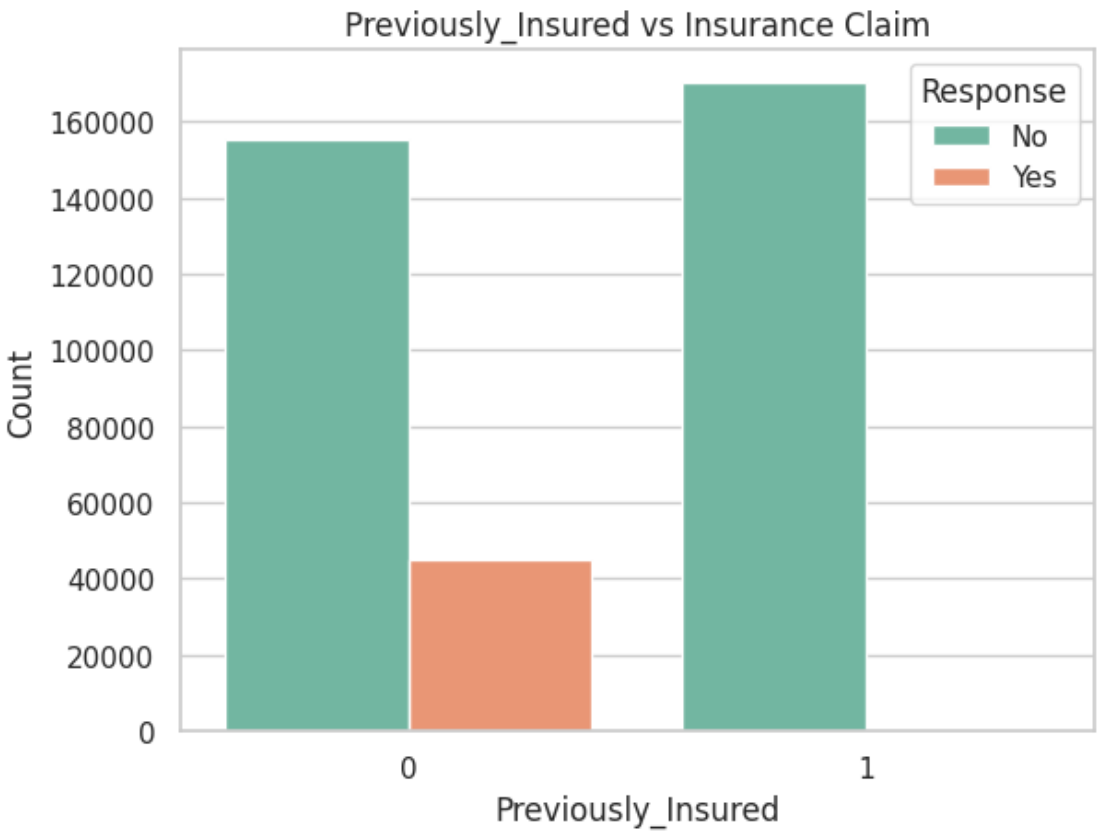
[ ]:
```
claim_rate_by_vehicle_age
```

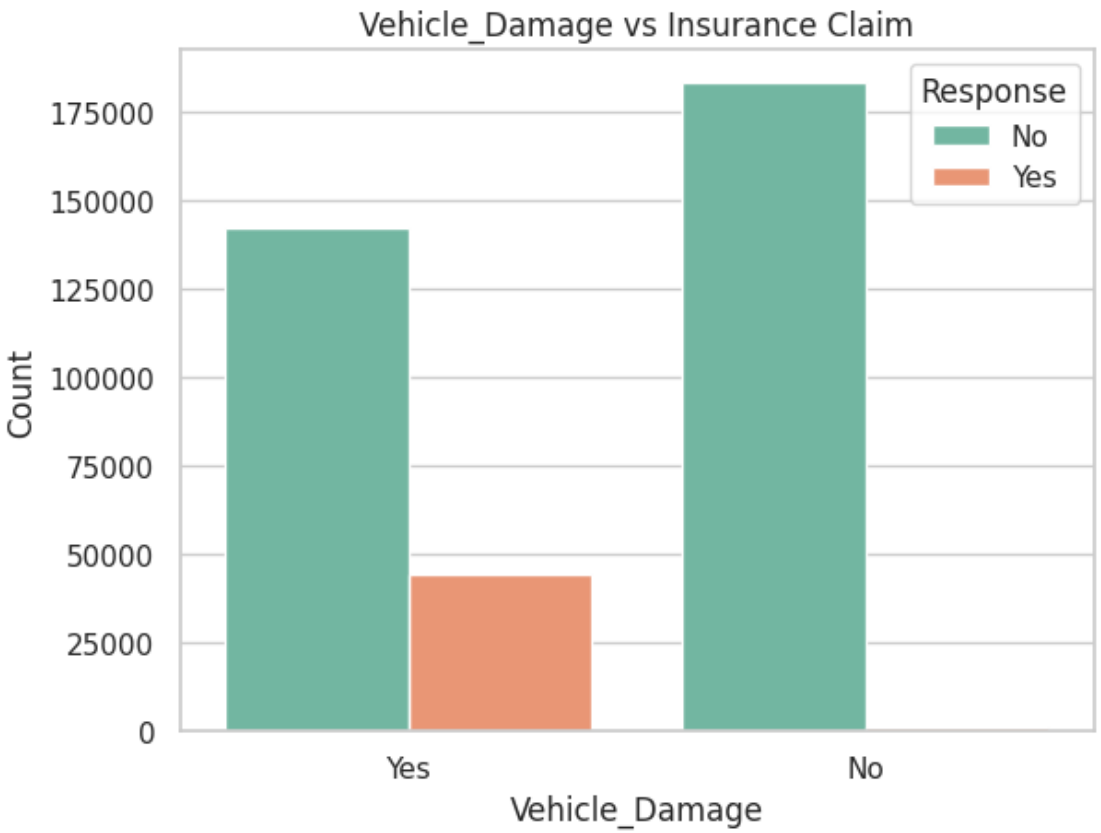
[ ]:
```
Vehicle_Age
1-2 Year     0.173753
< 1 Year     0.043702
> 2 Years    0.289421
Name: Response, dtype: float64
```

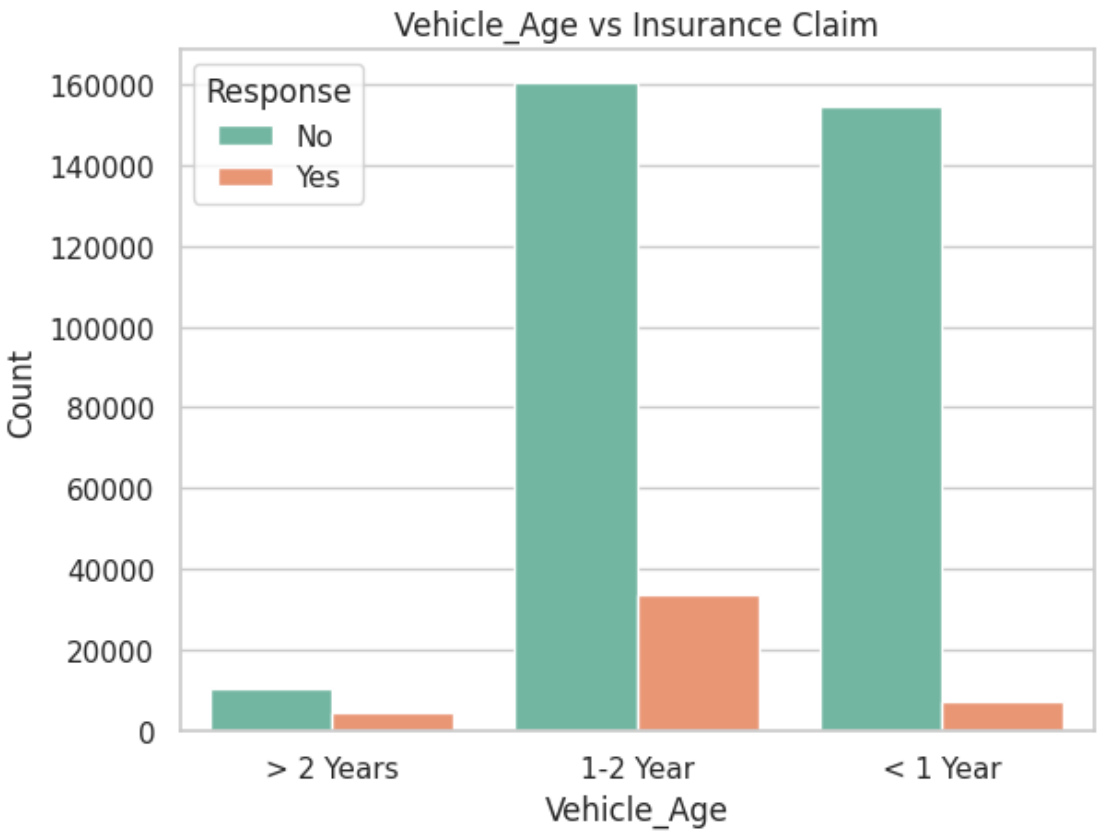
[ ]:
```
cat_factors = ['Gender', 'Previously_Insured', 'Vehicle_Damage', 'Vehicle_Age']
```

[ ]:
```
for feature in cat_factors:
    sns.countplot(x=feature, hue='Response', data=data, palette='Set2')
    plt.title(f'{feature} vs Insurance Claim')
    plt.xlabel(feature)
    plt.ylabel("Count")
    plt.legend(title='Response', labels=['No', 'Yes'])
    plt.show()
```


Gender vs Insurance Claim
Response
No
Yes
175000
150000
125000
100000
75000
50000
25000
0
Count
Male
Female
Gender


Previously_Insured vs Insurance Claim
Response
No
Yes
160000
140000
120000
100000
80000
60000
40000
20000
0
Count
0
1
Previously_Insured


Vehicle_Damage vs Insurance Claim
Response
No
Yes
175000
150000
125000
100000
75000
50000
25000
0
Count
Yes
No
Vehicle_Damage



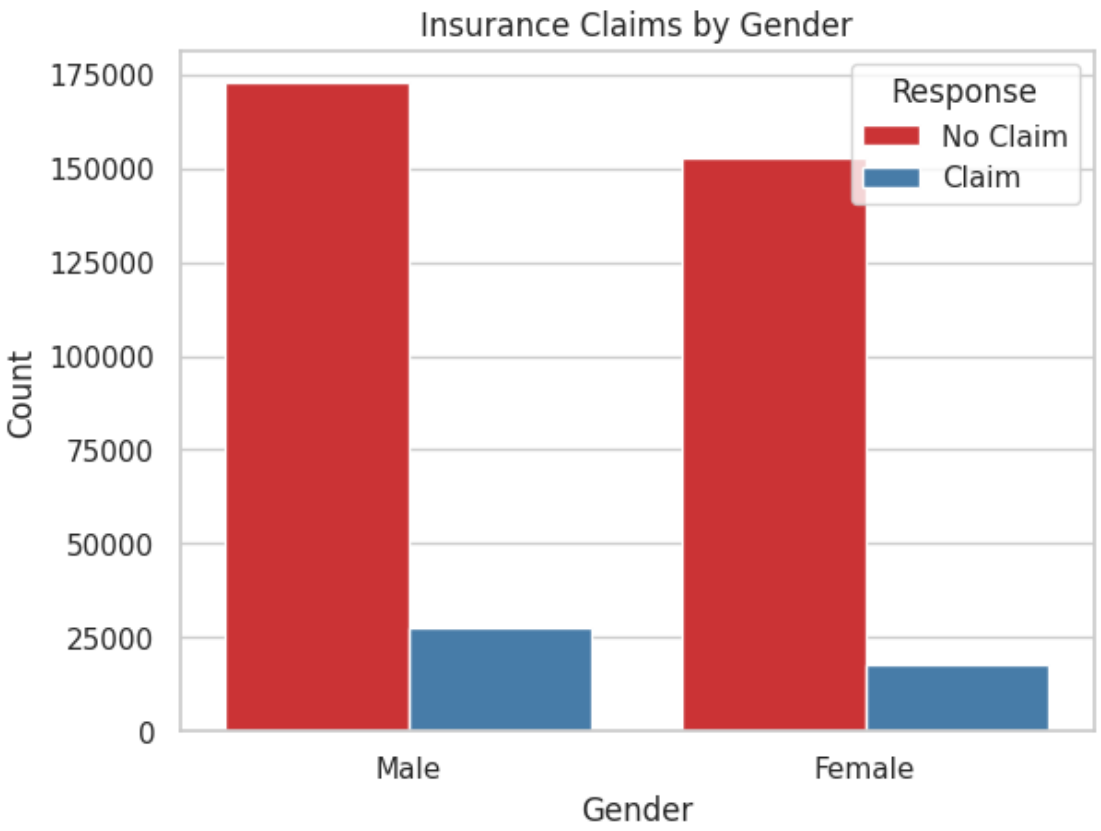
##8. Gender Analysis

[ ]:
```
gender_claim_rate = data.groupby('Gender')['Response'].mean()
```

[ ]:
```
gender_claim_rate
```

[ ]:
```
Gender
Female    0.103238
Male      0.137561
Name: Response, dtype: float64
```

[ ]:
```
sns.countplot(x='Gender', hue='Response', data=data, palette='Set1')
plt.title("Insurance Claims by Gender")
plt.xlabel("Gender")
plt.ylabel("Count")
plt.legend(title='Response', labels=['No Claim', 'Claim'])
plt.show()
```



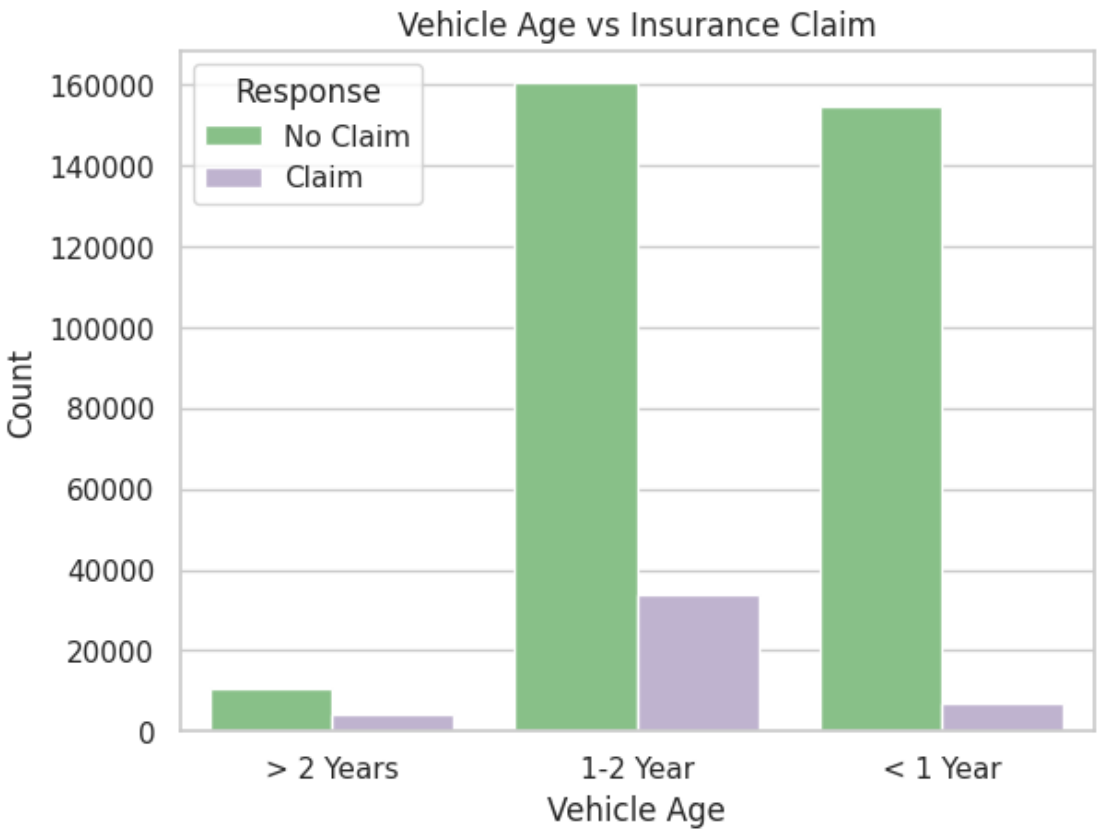
##9. Vehicle Age and Claims

[ ]:
```
vehicle_age_claim_rate = data.groupby('Vehicle_Age')['Response'].mean()
```

[ ]:
```
vehicle_age_claim_rate
```

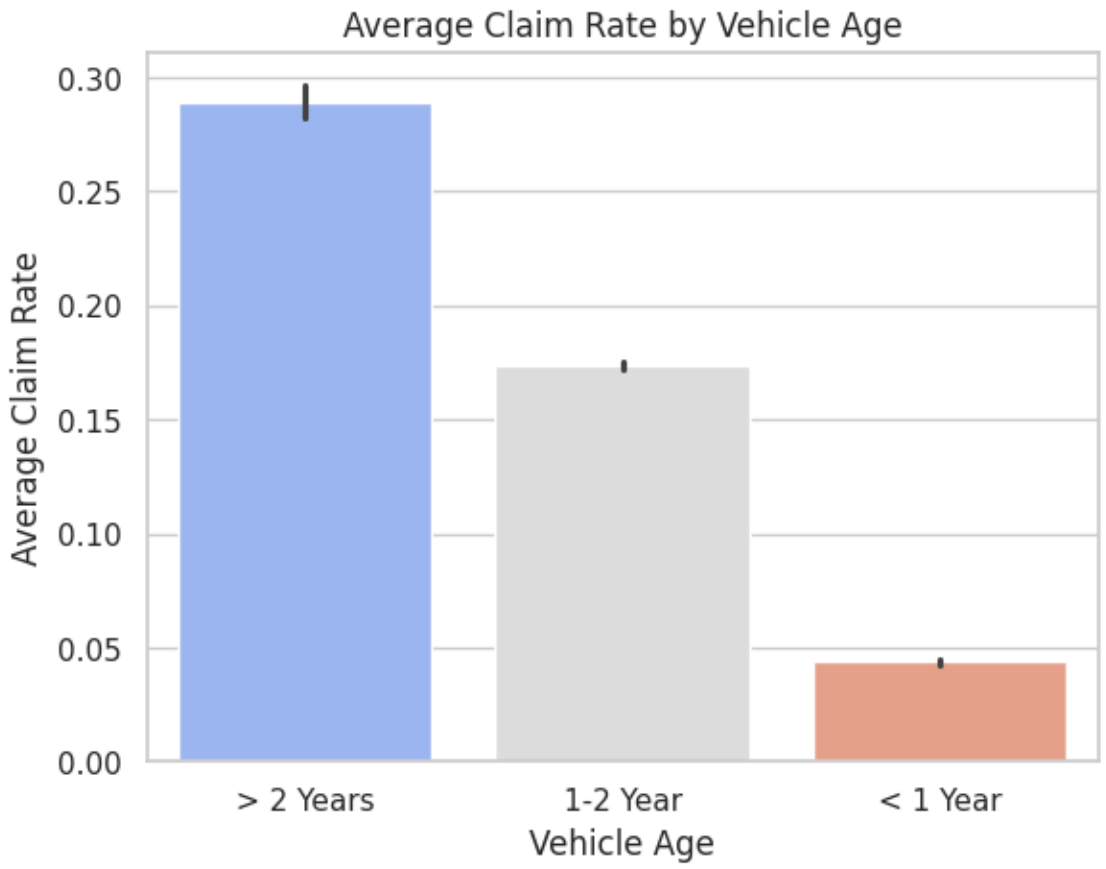
[ ]:
```
Vehicle_Age
1-2 Year     0.173753
< 1 Year     0.043702
> 2 Years    0.289421
Name: Response, dtype: float64
```

[ ]:
```
sns.countplot(x='Vehicle_Age', hue='Response', data=data, palette='Accent')
plt.title("Vehicle Age vs Insurance Claim")
plt.xlabel("Vehicle Age")
plt.ylabel("Count")
plt.legend(title='Response', labels=['No Claim', 'Claim'])
plt.show()
```



[ ]:
```
sns.barplot(x='Vehicle_Age', y='Response', data=data, estimator='mean',␣
 ↪palette='coolwarm')
plt.title("Average Claim Rate by Vehicle Age")
plt.xlabel("Vehicle Age")
plt.ylabel("Average Claim Rate")
plt.show()
```

```
<ipython-input-78-6fb564e8f6fd>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x='Vehicle_Age', y='Response', data=data, estimator='mean',
palette='coolwarm')
```



##10. Region Wise Analysis

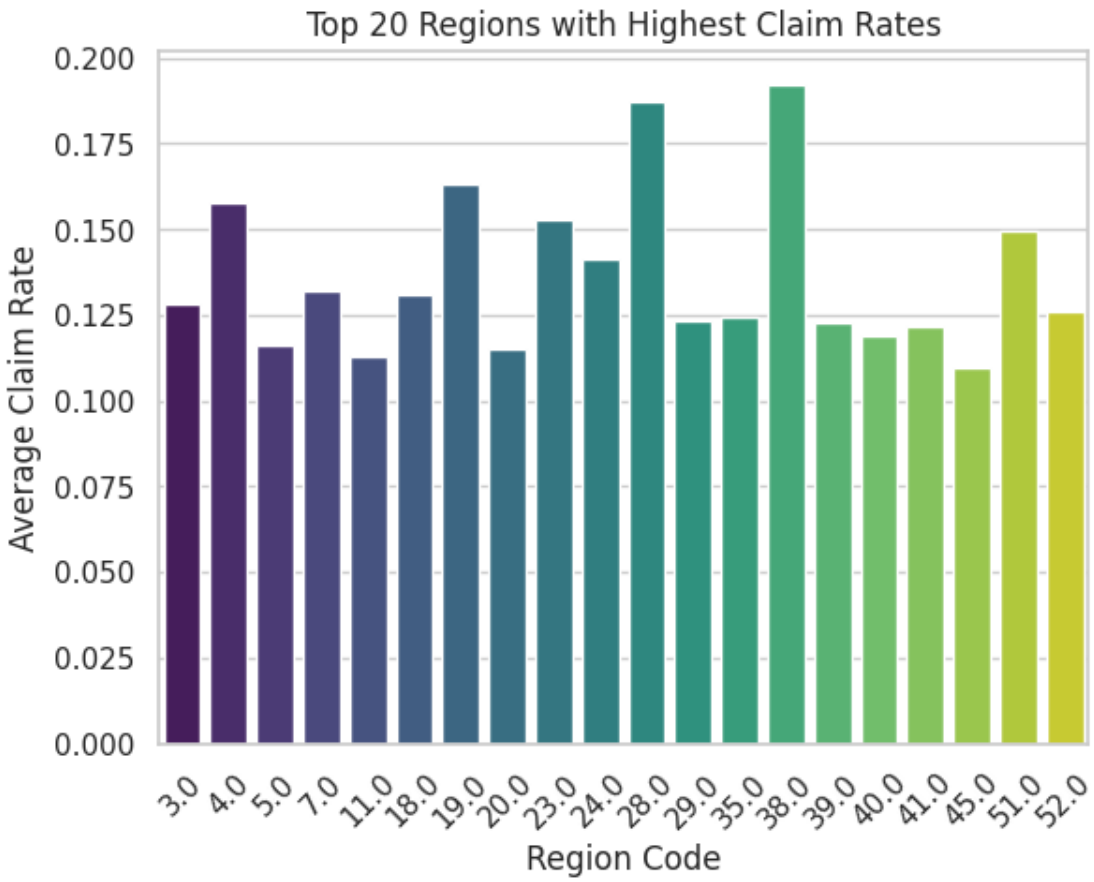
[ ]:
```
region_claim_rate = data.groupby('Region_Code')['Response'].mean().
↪sort_values(ascending=False)
```

[ ]:
```
region_claim_rate.head(10)
```

[ ]:
```
Region_Code
38.0    0.192423
28.0    0.187526
19.0    0.162973
4.0     0.157572
23.0    0.152707
51.0    0.149425
24.0    0.141611
7.0     0.132260
18.0    0.130987
3.0     0.128325
Name: Response, dtype: float64
```

[ ]:
```
region_avg = data.groupby('Region_Code')['Response'].mean().
  ↪sort_values(ascending=False).head(20)
sns.barplot(x=region_avg.index, y=region_avg.values, palette='viridis')
plt.title('Top 20 Regions with Highest Claim Rates')
plt.xlabel('Region Code')
plt.ylabel('Average Claim Rate')
plt.xticks(rotation=45)
plt.show()
```

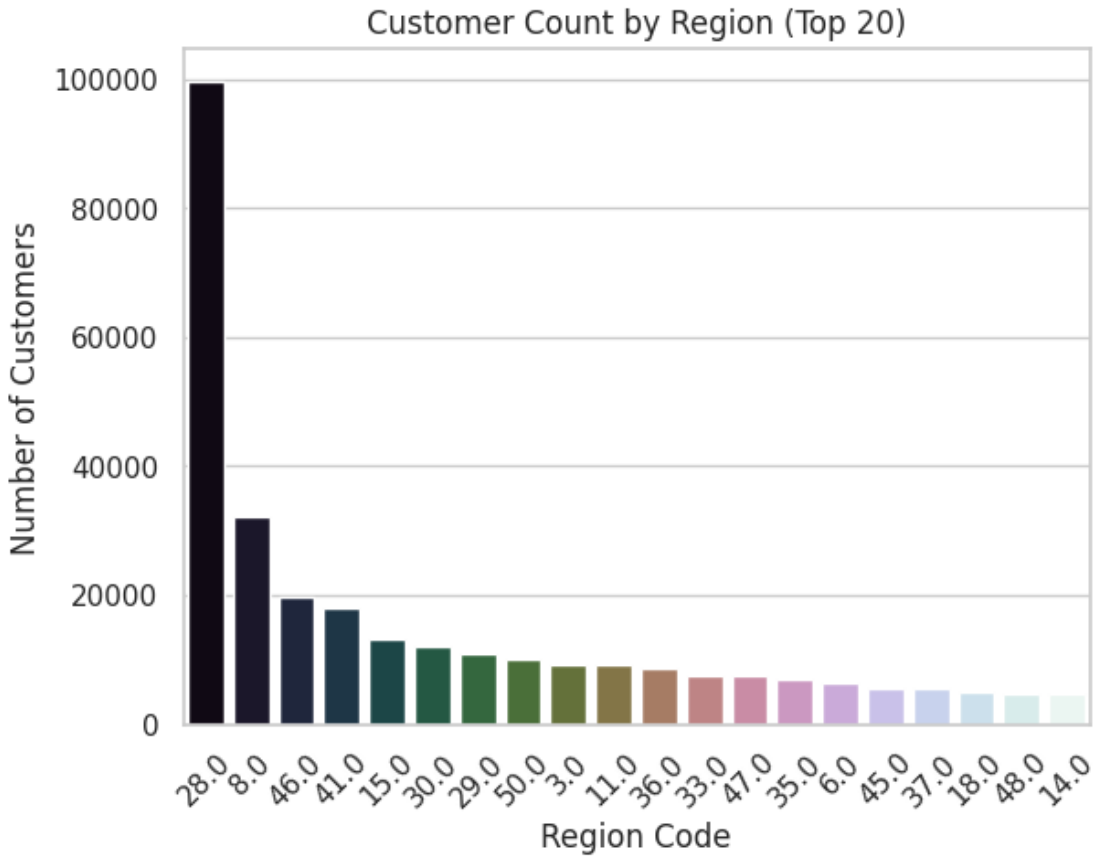
```
<ipython-input-54-8f44da0a10e4>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=region_avg.index, y=region_avg.values, palette='viridis')
```



[ ]:
```
sns.countplot(x='Region_Code', data=data, order=data['Region_Code'].
  ↪value_counts().index[:20], palette='cubehelix')
```

```
plt.title("Customer Count by Region (Top 20)")
plt.xlabel("Region Code")
plt.ylabel("Number of Customers")
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-56-e90f753a5892>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.countplot(x='Region_Code', data=data,
order=data['Region_Code'].value_counts().index[:20], palette='cubehelix')
```



##11. Policy Sales Channel Analysis

[ ]:
```
top_channels = data['Policy_Sales_Channel'].value_counts().head(10)
```

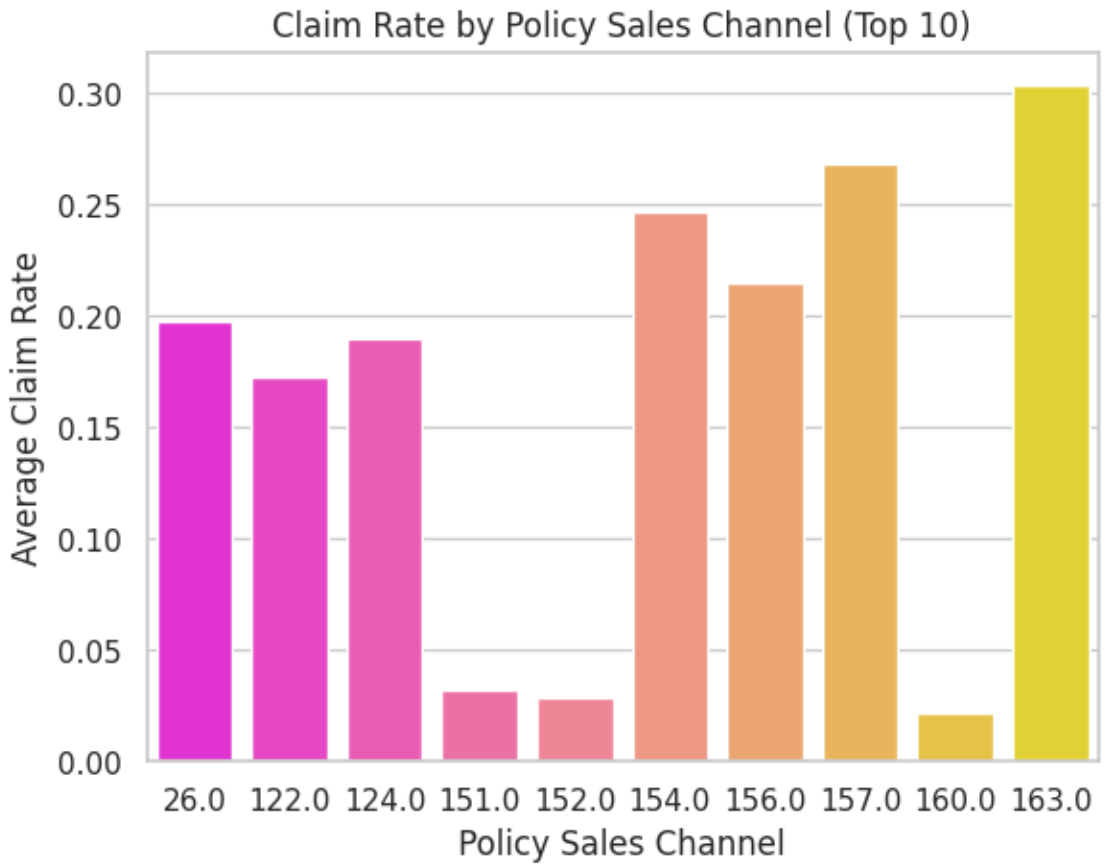
[ ]:
```
top_channels
```

[ ]:
```
Policy_Sales_Channel
152.0    132168
26.0      76375
124.0     71902
160.0     21489
156.0     10591
122.0      9306
157.0      6640
154.0      5890
151.0      3810
163.0      2856
Name: count, dtype: int64
```

[ ]:
```
top_channel_ids = top_channels.index
channel_claim_rate = data[data['Policy_Sales_Channel'].isin(top_channel_ids)].
 ↪groupby('Policy_Sales_Channel')['Response'].mean()
```

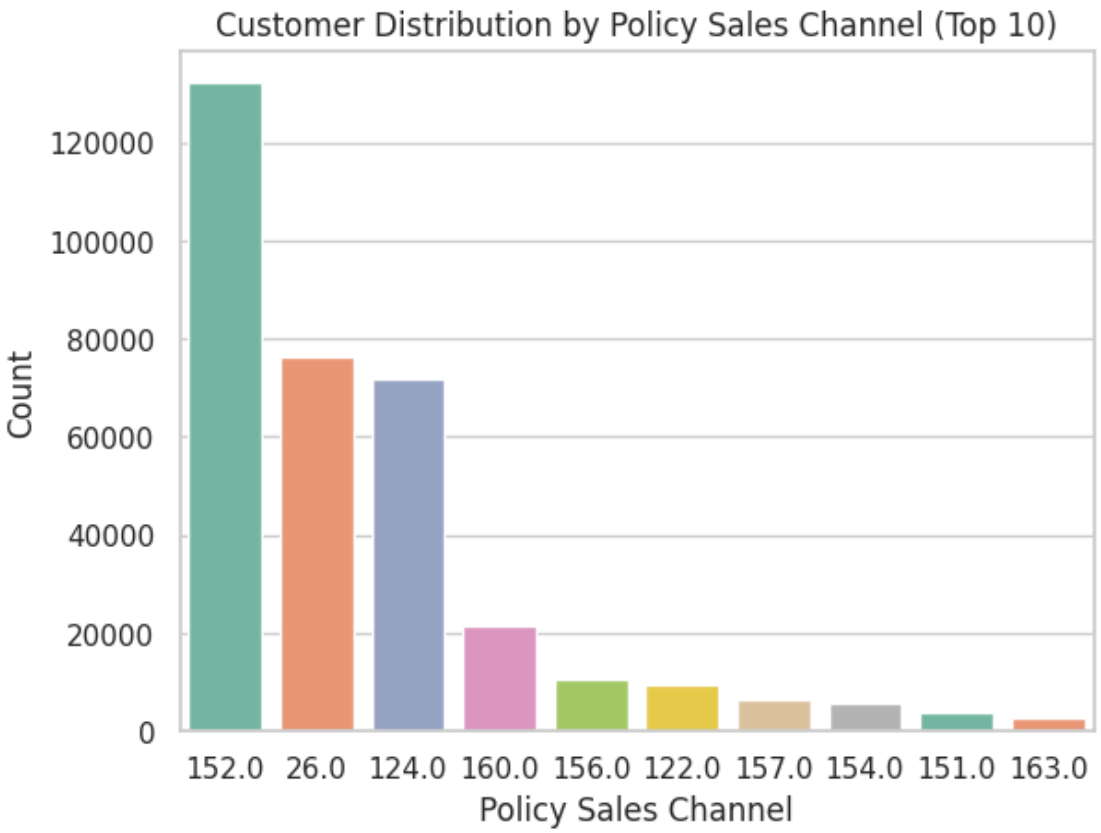
[ ]:
```
sns.barplot(x=channel_claim_rate.index, y=channel_claim_rate.values,␣
 ↪palette='spring')
plt.title('Claim Rate by Policy Sales Channel (Top 10)')
plt.xlabel('Policy Sales Channel')
plt.ylabel('Average Claim Rate')
plt.show()
```

```
<ipython-input-60-4a4393607f90>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=channel_claim_rate.index, y=channel_claim_rate.values,
palette='spring')
```



[ ]:
```
sns.countplot(x='Policy_Sales_Channel', data=data[data['Policy_Sales_Channel'].
  ↪isin(top_channel_ids)],
              order=top_channel_ids, palette='Set2')
plt.title('Customer Distribution by Policy Sales Channel (Top 10)')
plt.xlabel('Policy Sales Channel')
plt.ylabel('Count')
plt.show()
```

```
<ipython-input-61-0745a5b322e9>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.countplot(x='Policy_Sales_Channel',
data=data[data['Policy_Sales_Channel'].isin(top_channel_ids)],
```



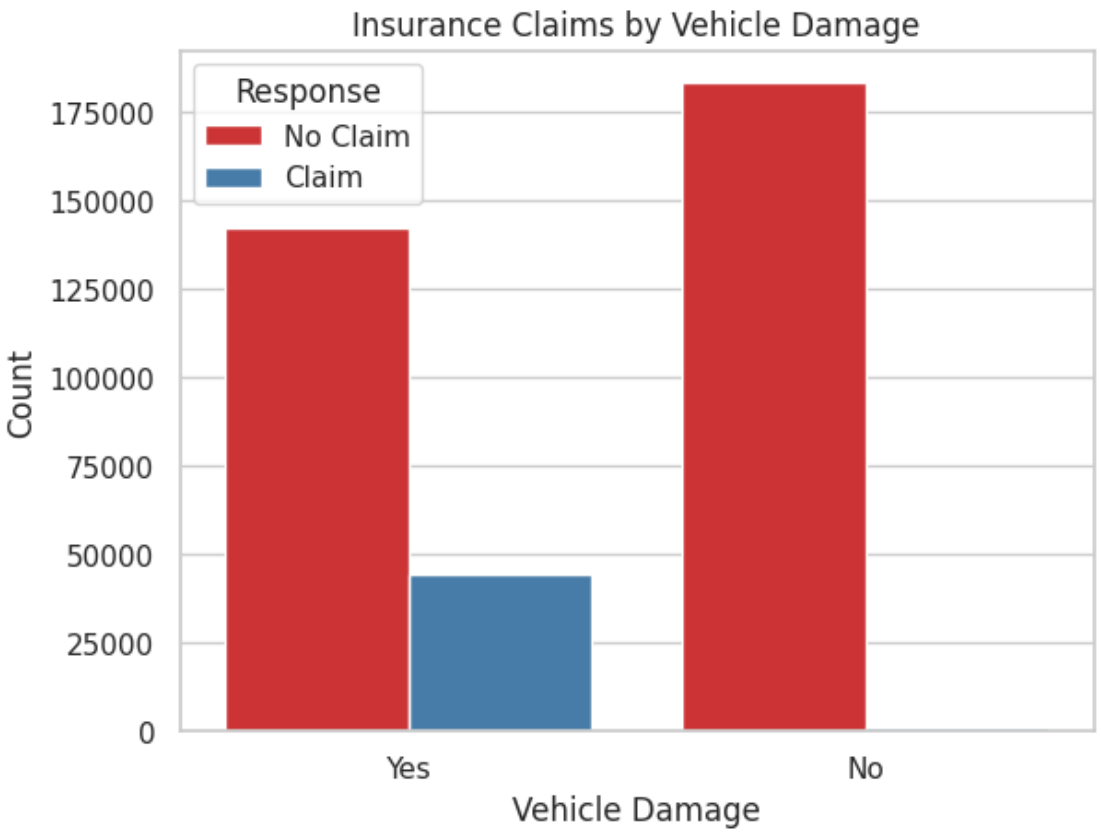
##12. Claim Frequency and Vehicle Damage

[ ]:
```
damage_claim_rate = data.groupby('Vehicle_Damage')['Response'].mean()
```

[ ]:
```
damage_claim_rate
```

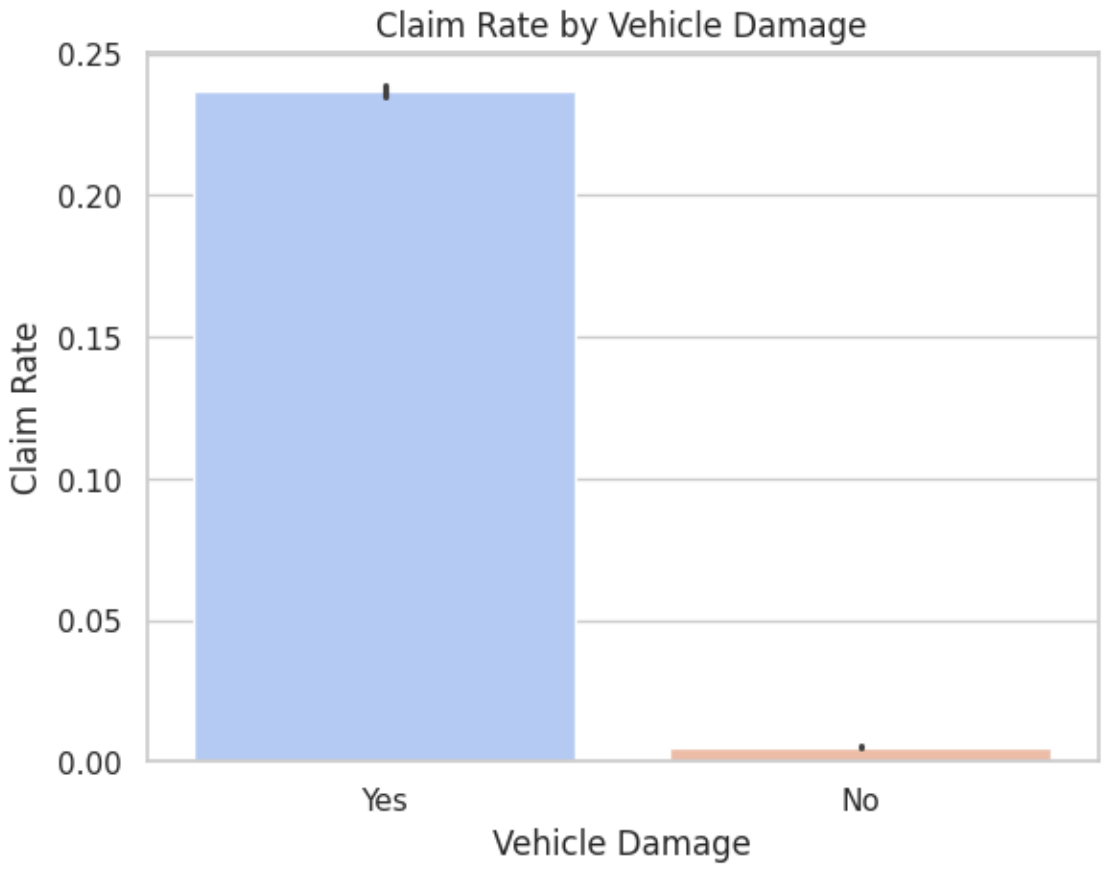
[ ]:
```
Vehicle_Damage
No     0.005249
Yes    0.236856
Name: Response, dtype: float64
```

[ ]:
```
sns.countplot(x='Vehicle_Damage', hue='Response', data=data, palette='Set1')
plt.title("Insurance Claims by Vehicle Damage")
plt.xlabel("Vehicle Damage")
plt.ylabel("Count")
plt.legend(title='Response', labels=['No Claim', 'Claim'])
plt.show()
```



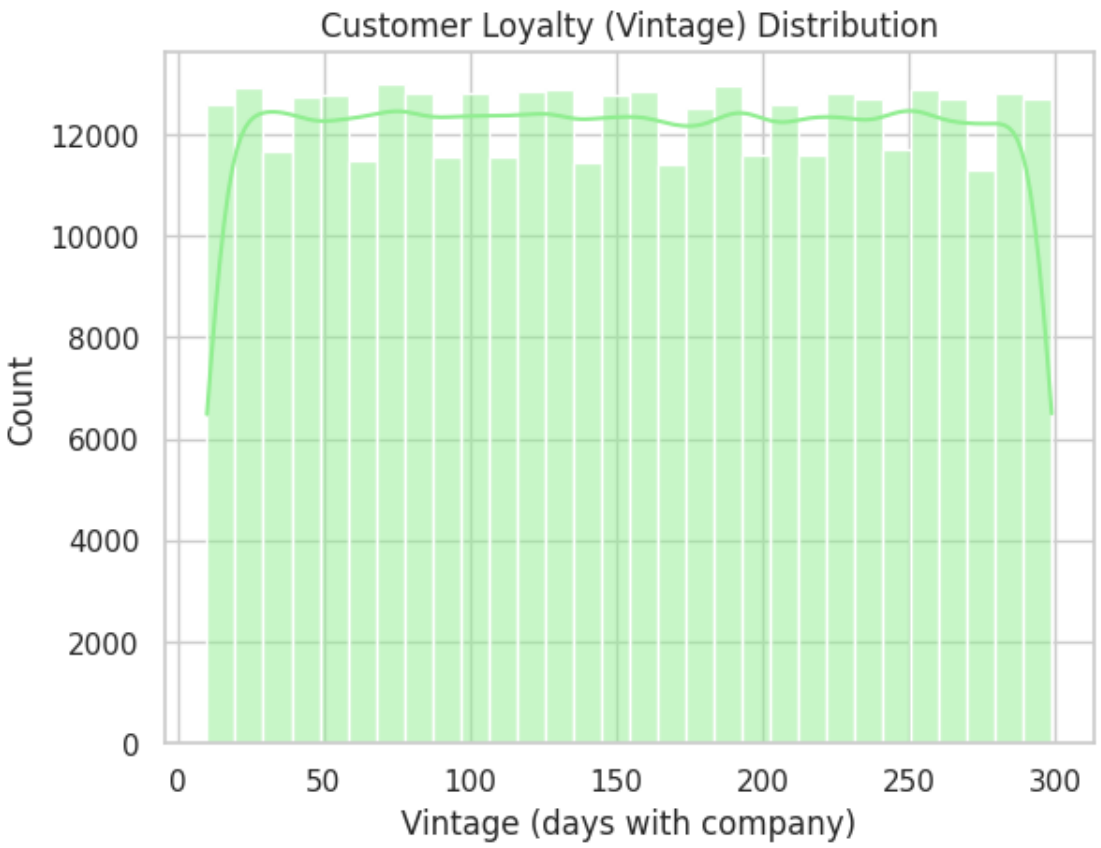
[ ]:
```
sns.barplot(x='Vehicle_Damage', y='Response', data=data, estimator='mean',␣
 ↪palette='coolwarm')
plt.title("Claim Rate by Vehicle Damage")
plt.xlabel("Vehicle Damage")
plt.ylabel("Claim Rate")
plt.show()
```

```
<ipython-input-65-b2c0636f03e6>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x='Vehicle_Damage', y='Response', data=data, estimator='mean',
palette='coolwarm')
```



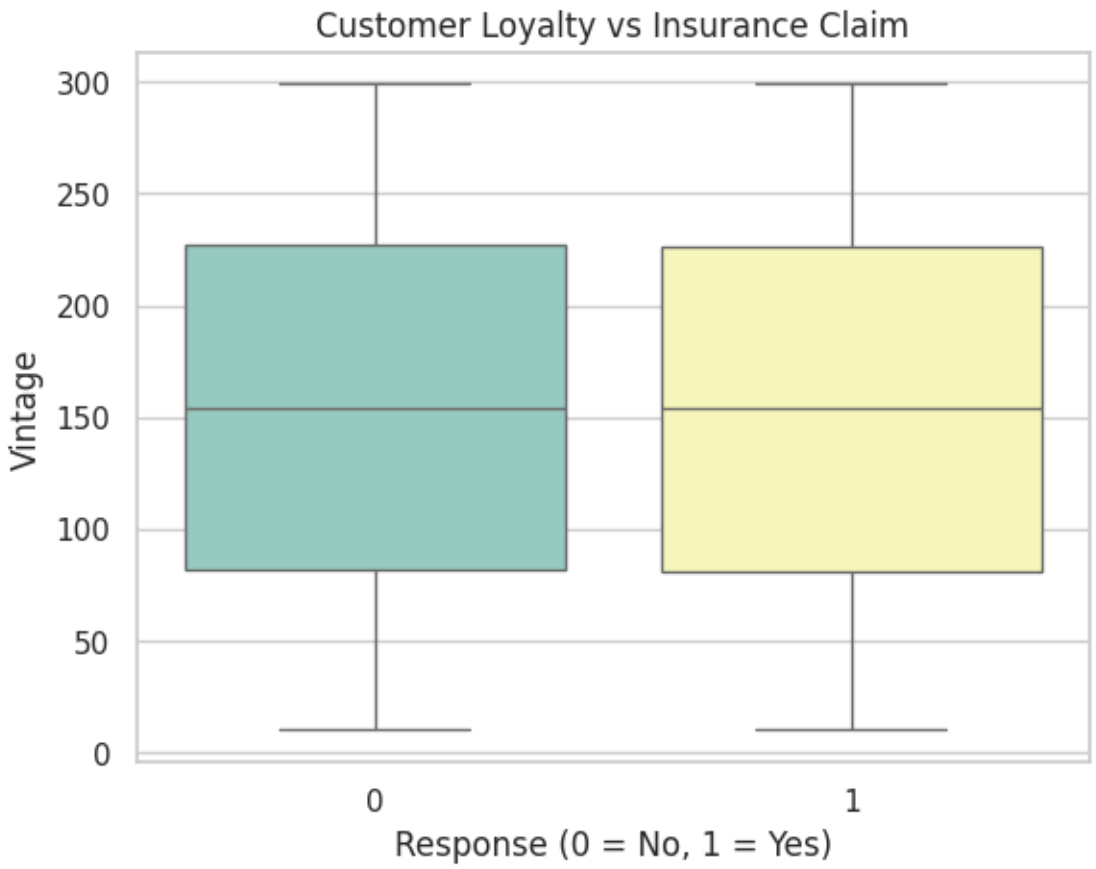
##13. Customer Loyalty (Vintage)

[ ]:
```
sns.histplot(data['Vintage'], bins=30, kde=True, color='lightgreen')
plt.title("Customer Loyalty (Vintage) Distribution")
plt.xlabel("Vintage (days with company)")
plt.ylabel("Count")
plt.show()
```



[ ]:
```
sns.boxplot(x='Response', y='Vintage', data=data, palette='Set3')
plt.title("Customer Loyalty vs Insurance Claim")
plt.xlabel("Response (0 = No, 1 = Yes)")
plt.ylabel("Vintage")
plt.show()
```

```
<ipython-input-67-49e99c93bfdc>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.boxplot(x='Response', y='Vintage', data=data, palette='Set3')
```



[ ]:
```
mean_vintage = data.groupby('Response')['Vintage'].mean()
```

[ ]:
```
mean_vintage
```

[ ]:
```
Response
0    154.396261
1    153.978961
Name: Vintage, dtype: float64
```

##14. Time Analysis(if applicable) ###Assuming 'vintage' is time related (grouped analysis)

[ ]:
```
data.dtypes
```

[ ]:
```
id                      int64
Gender                 object
Age                     int64
Driving_License         int64
Region_Code           float64
Previously_Insured      int64
Vehicle_Age            object
```

```
Vehicle_Damage           object
Annual_Premium          float64
Policy_Sales_Channel    float64
Vintage                   int64
Response                  int64
dtype: object
```

[ ]:
```
data['Vintage_Group'] = pd.qcut(data['Vintage'], q=4, labels=['Low', 'Medium',␣
↪'High', 'Very High'])
```

[ ]:
```
vintage_claim_rate = data.groupby('Vintage_Group')['Response'].mean()
```

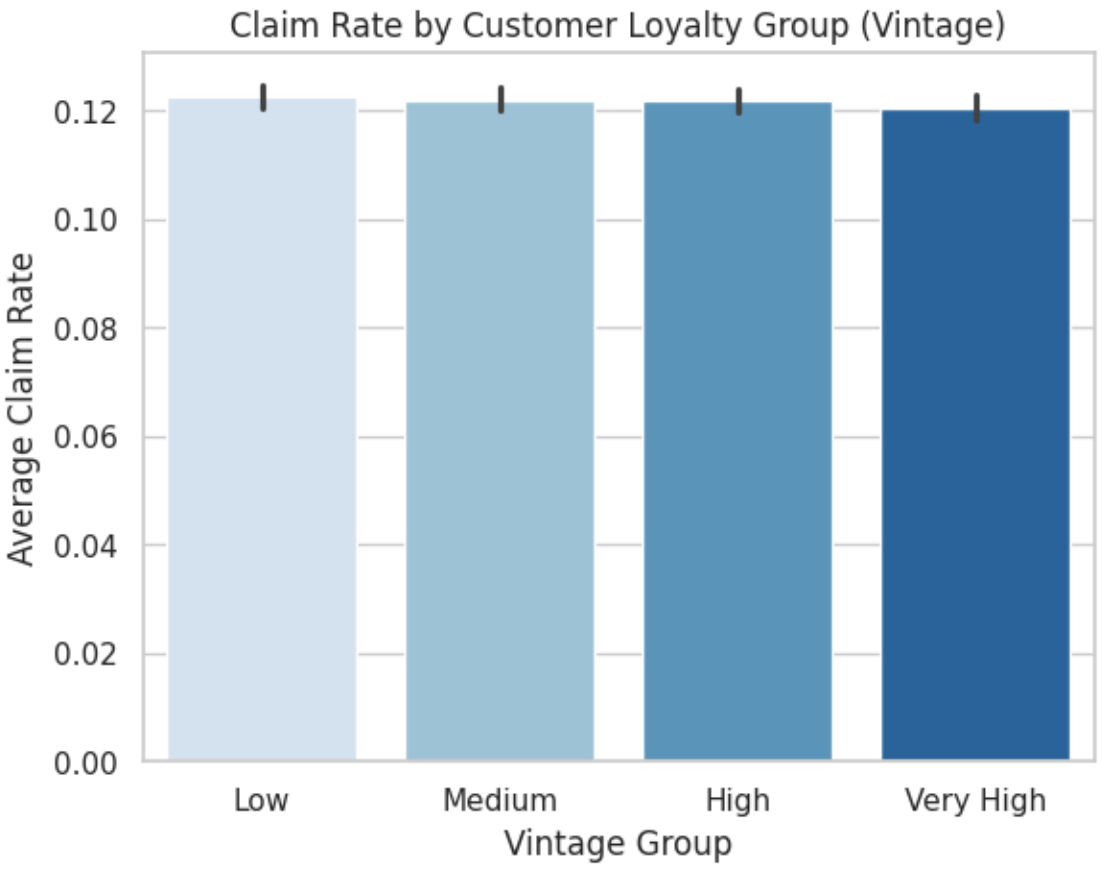
```
<ipython-input-74-d5d59ba37081>:1: FutureWarning: The default of observed=False
is deprecated and will be changed to True in a future version of pandas. Pass
observed=False to retain current behavior or observed=True to adopt the future
default and silence this warning.
  vintage_claim_rate = data.groupby('Vintage_Group')['Response'].mean()
```

[ ]:
```
vintage_claim_rate
```

[ ]:
```
Vintage_Group
Low          0.122637
Medium       0.122001
High         0.121953
Very High    0.120513
Name: Response, dtype: float64
```

[ ]:
```
sns.barplot(x='Vintage_Group', y='Response', data=data, estimator='mean',␣
↪palette='Blues')
plt.title("Claim Rate by Customer Loyalty Group (Vintage)")
plt.xlabel("Vintage Group")
plt.ylabel("Average Claim Rate")
plt.show()
```

```
<ipython-input-77-81d52615ea02>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x='Vintage_Group', y='Response', data=data, estimator='mean',
palette='Blues')
```


Claim Rate by Customer Loyalty Group (Vintage)
Average Claim Rate
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Low
Medium
High
Very High
Vintage Group