## 🎯 ***Project Objective***: To provide personalized vehicle insurance offerings by analyzing customer profiles, vehicle attributes, and claim likelihood, improving customer targeting and optimizing underwriting processes.

### 📊 Data-Driven Foundations The uploaded dataset includes essential features like:

Customer Profile: Age, Gender, Region Code

Vehicle Details: Vehicle Age, Vehicle Damage, Previously Insured

Insurance Factors: Policy Sales Channel, Annual Premium, Response (claim interest)

### 🧠 ***Key Insights for Startup Strategy***

1. Customer Segmentation Segment users by age, vehicle damage history, and insurance history to tailor insurance products.

Example: Younger drivers with prior damage might be offered different premiums than older, safe drivers.

1. Risk Profiling & Underwriting Use features like Previously_Insured, Vehicle_Age, and Vehicle_Damage to predict risk and set premium tiers accordingly.

1. Predictive Modeling Target customers likely to purchase insurance (based on Response).

2. Regional Market Strategy Analyze Region_Code distribution to identify high-demand regions for targeted campaigns or regional partnerships.

3. Digital Distribution Evaluate the effectiveness of various Policy_Sales_Channel platforms.

Focus on digital-first channels to reduce customer acquisition cost.

- Import Libraries

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

## *Step 1. Data Loading and Inspection:*

```
data=pd.read_csv("Vehicle_Insurance.csv")
data
```

| | id | Gender | Age | Driving_License | Region_Code | Previously_Insured | Vehicle_Age | Vehicle_Damage | Annual_Premium | Policy_Sales_Channel | Vintage | Response |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | Male | 44 | 1 | 28.0 | 0 | > 2 Years | Yes | 40454.0 | 26.0 | 217 | 1 |
| **1** | 2 | Male | 76 | 1 | 3.0 | 0 | 1-2 Year | No | 33536.0 | 26.0 | 183 | 0 |
| **2** | 3 | Male | 47 | 1 | 28.0 | 0 | > 2 Years | Yes | 38294.0 | 26.0 | 27 | 1 |
| **3** | 4 | Male | 21 | 1 | 11.0 | 1 | < 1 Year | No | 28619.0 | 152.0 | 203 | 0 |
| **4** | 5 | Female | 29 | 1 | 41.0 | 1 | < 1 Year | No | 27496.0 | 152.0 | 39 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **381104** | 381105 | Male | 74 | 1 | 26.0 | 1 | 1-2 Year | No | 30170.0 | 26.0 | 88 | 0 |
| **381105** | 381106 | Male | 30 | 1 | 37.0 | 1 | < 1 Year | No | 40016.0 | 152.0 | 131 | 0 |
| **381106** | 381107 | Male | 21 | 1 | 30.0 | 1 | < 1 Year | No | 35118.0 | 160.0 | 161 | 0 |
| **381107** | 381108 | Female | 68 | 1 | 14.0 | 0 | > 2 Years | Yes | 44617.0 | 124.0 | 74 | 0 |
| **381108** | 381109 | Male | 46 | 1 | 29.0 | 0 | 1-2 Year | No | 41777.0 | 26.0 | 237 | 0 |

381109 rows × 12 columns

{"type":"dataframe","variable_name":"data"}

```
data.head(12)
```

| | id | Gender | Age | Driving_License | Region_Code | Previously_Insured | Vehicle_Age | Vehicle_Damage | Annual_Premium | Policy_Sales_Channel | Vintage | Response |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | Male | 44 | 1 | 28.0 | 0 | > 2 Years | Yes | 40454.0 | 26.0 | 217 | 1 |
| **1** | 2 | Male | 76 | 1 | 3.0 | 0 | 1-2 Year | No | 33536.0 | 26.0 | 183 | 0 |
| **2** | 3 | Male | 47 | 1 | 28.0 | 0 | > 2 Years | Yes | 38294.0 | 26.0 | 27 | 1 |
| **3** | 4 | Male | 21 | 1 | 11.0 | 1 | < 1 Year | No | 28619.0 | 152.0 | 203 | 0 |
| **4** | 5 | Female | 29 | 1 | 41.0 | 1 | < 1 Year | No | 27496.0 | 152.0 | 39 | 0 |
| **5** | 6 | Female | 24 | 1 | 33.0 | 0 | < 1 Year | Yes | 2630.0 | 160.0 | 176 | 0 |
| **6** | 7 | Male | 23 | 1 | 11.0 | 0 | < 1 Year | Yes | 23367.0 | 152.0 | 249 | 0 |
| **7** | 8 | Female | 56 | 1 | 28.0 | 0 | 1-2 Year | Yes | 32031.0 | 26.0 | 72 | 1 |
| **8** | 9 | Female | 24 | 1 | 3.0 | 1 | < 1 Year | No | 27619.0 | 152.0 | 28 | 0 |
| **9** | 10 | Female | 32 | 1 | 6.0 | 1 | < 1 Year | No | 28771.0 | 152.0 | 80 | 0 |
| **10** | 11 | Female | 47 | 1 | 35.0 | 0 | 1-2 Year | Yes | 47576.0 | 124.0 | 46 | 1 |
| **11** | 12 | Female | 24 | 1 | 50.0 | 1 | < 1 Year | No | 48699.0 | 152.0 | 289 | 0 |

{"type":"dataframe","variable_name":"data"}

```
# Identifying the types of information available.
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381109 entries, 0 to 381108
Data columns (total 12 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   id                    381109 non-null  int64  
 1   Gender                381109 non-null  object 
 2   Age                   381109 non-null  int64  
 3   Driving_License       381109 non-null  int64  
 4   Region_Code           381109 non-null  float64
 5   Previously_Insured    381109 non-null  int64  
 6   Vehicle_Age           381109 non-null  object 
 7   Vehicle_Damage        381109 non-null  object 
 8   Annual_Premium        381109 non-null  float64
 9   Policy_Sales_Channel  381109 non-null  float64
 10  Vintage               381109 non-null  int64
```

```
 11  Response              381109 non-null  int64
dtypes: float64(3), int64(6), object(3)
memory usage: 34.9+ MB
```

```
data.describe()
```

| | id | Age | Driving_License | Region_Code | Previously_Insured | Annual_Premium | Policy_Sales_Channel | Vintage | Response |
|---|---|---|---|---|---|---|---|---|---|
| count | 381109.000000 | 381109.000000 | 381109.000000 | 381109.000000 | 381109.000000 | 381109.000000 | 381109.000000 | 381109.000000 | 381109.000000 |
| mean | 190555.000000 | 38.822584 | 0.997869 | 26.388807 | 0.458210 | 30564.389581 | 112.034295 | 154.347397 | 0.122563 |
| std | 110016.836208 | 15.511611 | 0.046110 | 13.229888 | 0.498251 | 17213.155057 | 54.203995 | 83.671304 | 0.327936 |
| min | 1.000000 | 20.000000 | 0.000000 | 0.000000 | 0.000000 | 2630.000000 | 1.000000 | 10.000000 | 0.000000 |
| 25% | 95278.000000 | 25.000000 | 1.000000 | 15.000000 | 0.000000 | 24405.000000 | 29.000000 | 82.000000 | 0.000000 |
| 50% | 190555.000000 | 36.000000 | 1.000000 | 28.000000 | 0.000000 | 31669.000000 | 133.000000 | 154.000000 | 0.000000 |
| 75% | 285832.000000 | 49.000000 | 1.000000 | 35.000000 | 1.000000 | 39400.000000 | 152.000000 | 227.000000 | 0.000000 |
| max | 381109.000000 | 85.000000 | 1.000000 | 52.000000 | 1.000000 | 540165.000000 | 163.000000 | 299.000000 | 1.000000 |

## ***Step 2. DATA CLEANING:***

checking the nan values,duplicated rows,converting decimal to integer,replacing catagorical to numerical data.

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

```
{"repr_error":"Out of range float values are not JSON compliant:
nan","type":"dataframe"}
```

```
data.duplicated().sum()
```

```
np.int64(0)
```

```
data["Region_Code"]=data["Region_Code"].astype("int64")
```

```
data["Gender"] = data["Gender"].replace({"Male": 0, "Female": 1})
data["Vehicle_Damage"] = data["Vehicle_Damage"].replace({"Yes": 0, "No": 1})
```

```
/tmp/ipython-input-11-2955536721.py:1: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain
the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-
in to the future behavior, set `pd.set_option('future.no_silent_downcasting',
True)`
  data["Gender"] = data["Gender"].replace({"Male": 0, "Female": 1})
/tmp/ipython-input-11-2955536721.py:2: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain
the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-
in to the future behavior, set `pd.set_option('future.no_silent_downcasting',
True)`
  data["Vehicle_Damage"] = data["Vehicle_Damage"].replace({"Yes": 0, "No":
1})
```

```
data.head()
```

| | id | Gender | Age | Driving_License | Region_Code | Previously_Insured | Vehicle_Age | Vehicle_Damage | Annual_Premium | Policy_Sales_Channel | Vintage | Response |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0 | 44 | 1 | 28 | 0 | > 2 Years | 0 | 40454.0 | 26.0 | 217 | 1 |
| **1** | 2 | 0 | 76 | 1 | 3 | 0 | 1-2 Year | 1 | 33536.0 | 26.0 | 183 | 0 |
| **2** | 3 | 0 | 47 | 1 | 28 | 0 | > 2 Years | 0 | 38294.0 | 26.0 | 27 | 1 |
| **3** | 4 | 0 | 21 | 1 | 11 | 1 | < 1 Year | 1 | 28619.0 | 152.0 | 203 | 0 |
| **4** | 5 | 1 | 29 | 1 | 41 | 1 | < 1 Year | 1 | 27496.0 | 152.0 | 39 | 0 |

{"type":"dataframe","variable_name":"data"}

```
# check the unique and removing extra space if present and checking the value
counts of "Vehicle_Age".
data["Vehicle_Age"].unique()
```

```
array(['> 2 Years', '1-2 Year', '< 1 Year'], dtype=object)
```

```
data["Vehicle_Age"].str.strip()
```

```
0         > 2 Years
1          1-2 Year
2         > 2 Years
3          < 1 Year
4          < 1 Year
           ...
381104     1-2 Year
381105     < 1 Year
381106     < 1 Year
381107    > 2 Years
381108     1-2 Year
Name: Vehicle_Age, Length: 381109, dtype: object
```

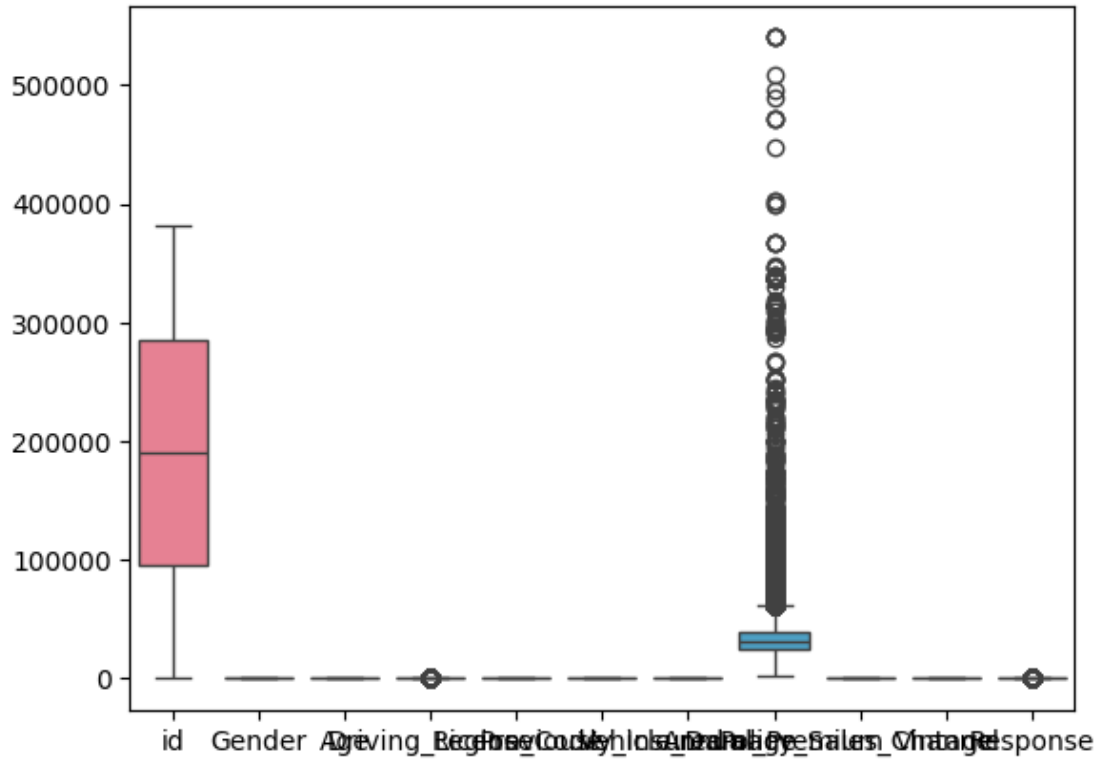
```
data["Vehicle_Age"].value_counts()
```

```
Vehicle_Age
1-2 Year     200316
< 1 Year     164786
```

```
> 2 Years     16007
Name: count, dtype: int64
```

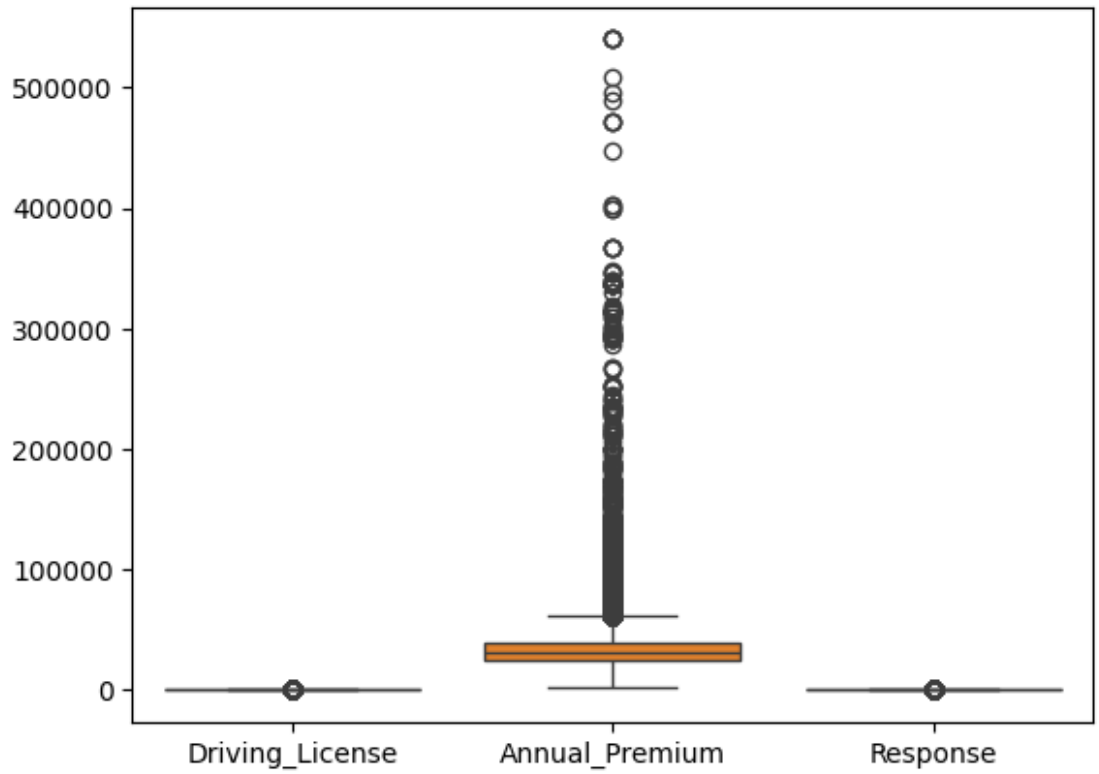
```
#Detecting the outliers
sns.boxplot(data=data)
```

```
<Axes: >
```




```
sns.boxplot(data=data[["Driving_License","Annual_Premium","Response"]])
```

```
<Axes: >
```




```
q1=data["Driving_License"].quantile(0.25)
q3=data["Driving_License"].quantile(0.75)
iqr=q3-q1
iqr
```

```
np.float64(0.0)
```

```
q1=data["Response"].quantile(0.25)
q3=data["Response"].quantile(0.75)
iqr=q3-q1
iqr
```

```
np.float64(0.0)
```

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

```
np.float64(14995.0)
```

```
lower_fence=Q1-1.5*IQR
upper_fence=Q3+1.5*IQR
```

```
lower_fence
```

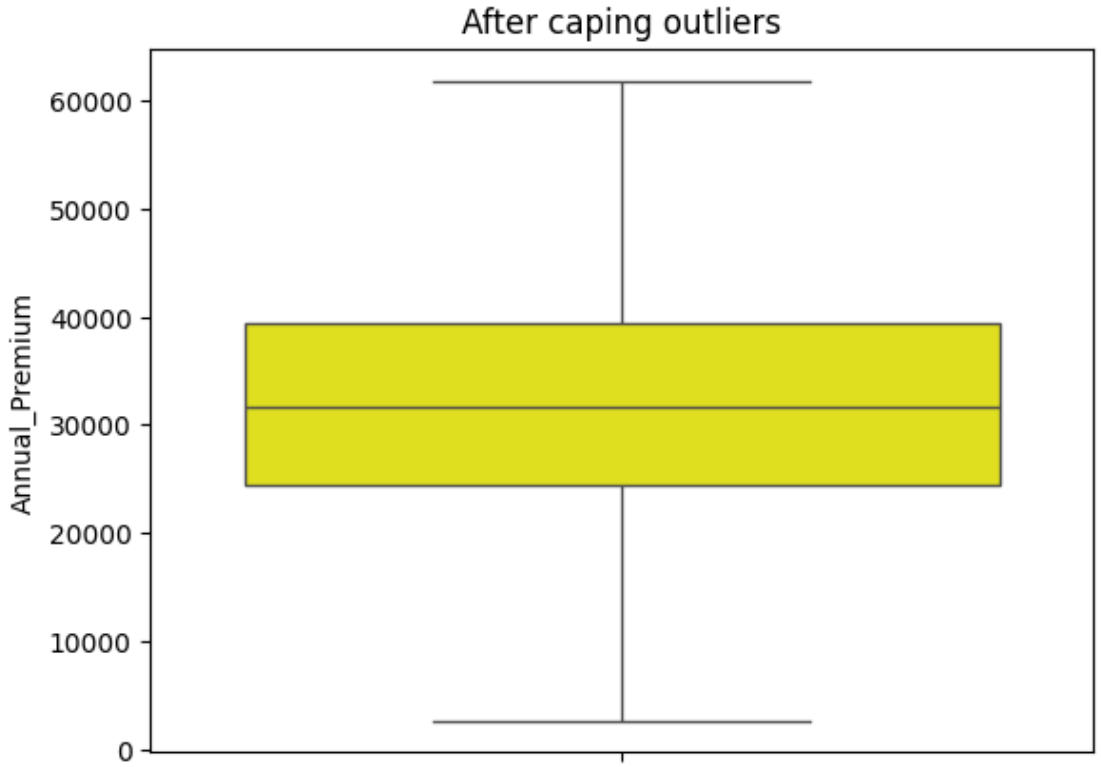
```
np.float64(1912.5)
```

```
upper_fence
```

```
np.float64(61892.5)
```

```
#Caping outliers
data["Annual_Premium"]=np.where(data["Annual_Premium"]<=lower_fence,lower_fen
ce,data["Annual_Premium"])
data["Annual_Premium"]=np.where(data["Annual_Premium"]>=upper_fence,upper_fen
ce,data["Annual_Premium"])
```

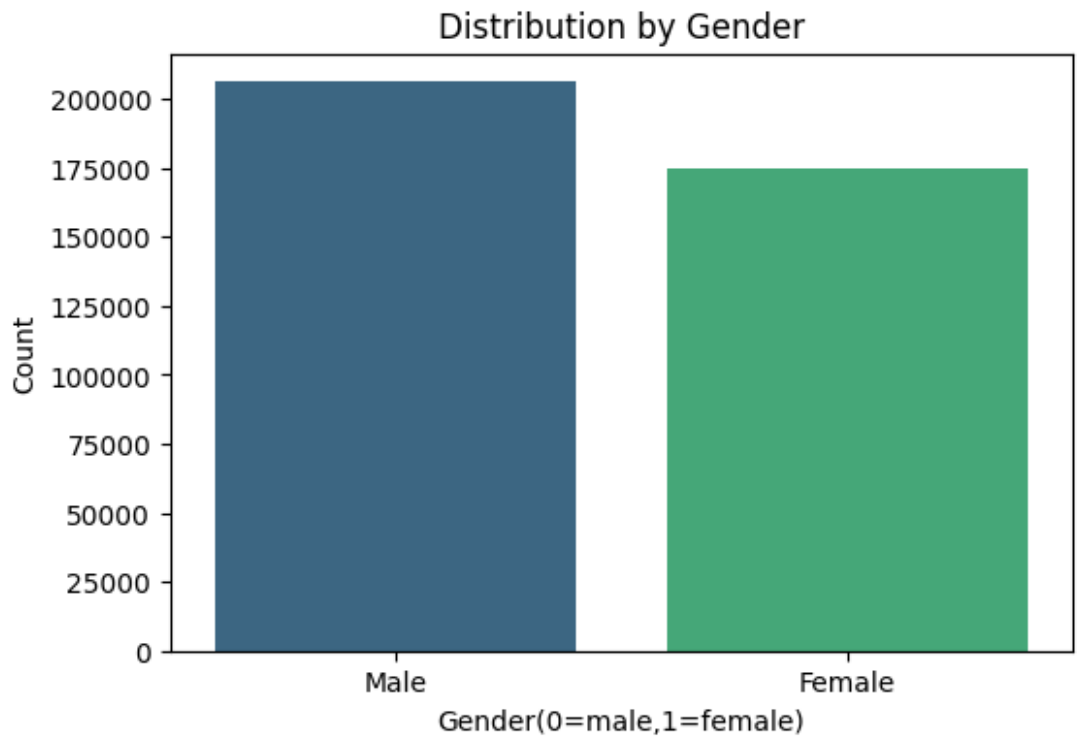
```
#Boxplot after caping
sns.boxplot(data["Annual_Premium"],color="yellow")
plt.title("After caping outliers")
plt.show()
```




## Step 3. Data Visualization:

Utilize various visualization techniques to explore the distribution of key variables.

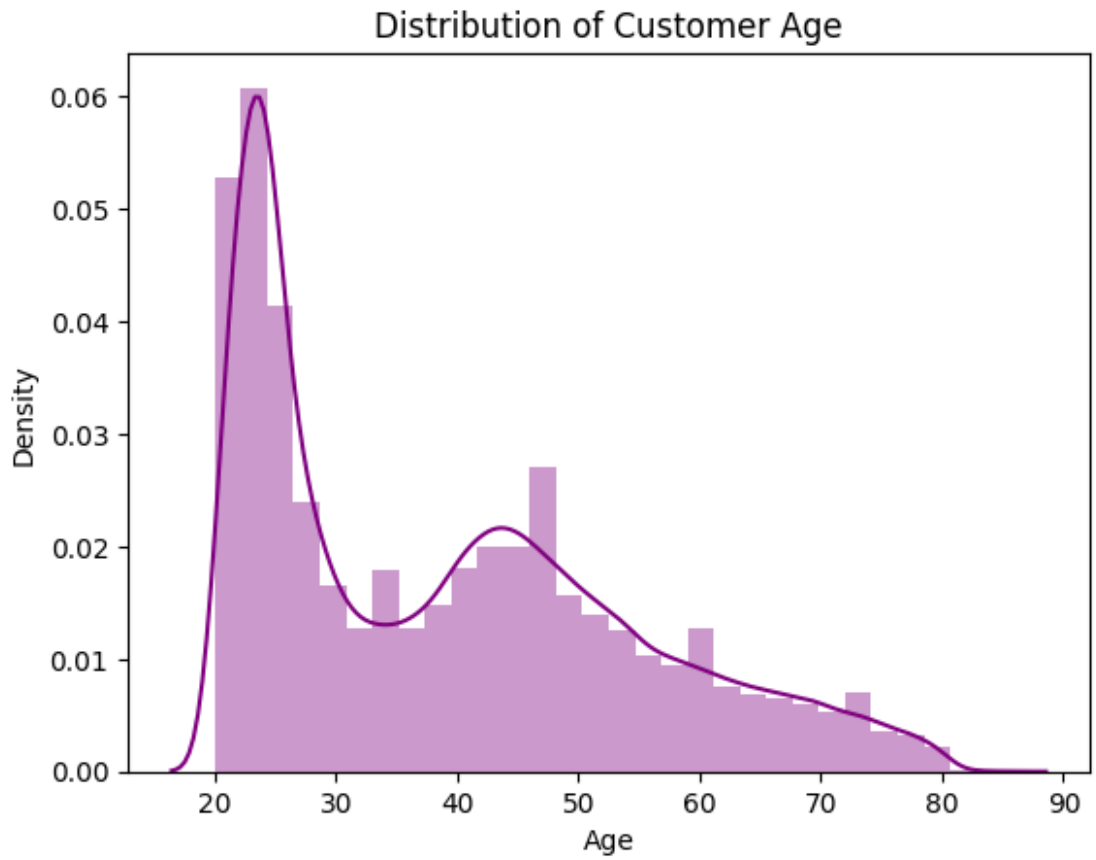
```
# plot1. Gender distribution using countplot
plt.figure(figsize=(6, 4))
sns.countplot(x='Gender', data=data, palette='viridis')
plt.title('Distribution by Gender')
plt.xlabel('Gender(0=male,1=female)')
plt.ylabel('Count')
plt.show()
```




✅ ***INSIGHTS OF GENDER DISTRIBUTION***: This could suggest that more men are involved in vehicle ownership or insurance inquiries compared to women, at least within the data sample.

```
#Plot 2. Age distribution using distplot
sns.distplot(data['Age'], bins=30, kde=True,color="purple")
plt.title('Distribution of Customer Age')
plt.xlabel('Age')
plt.show()
```

```
/tmp/ipython-input-30-1853960550.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Age'], bins=30, kde=True,color="purple")
```

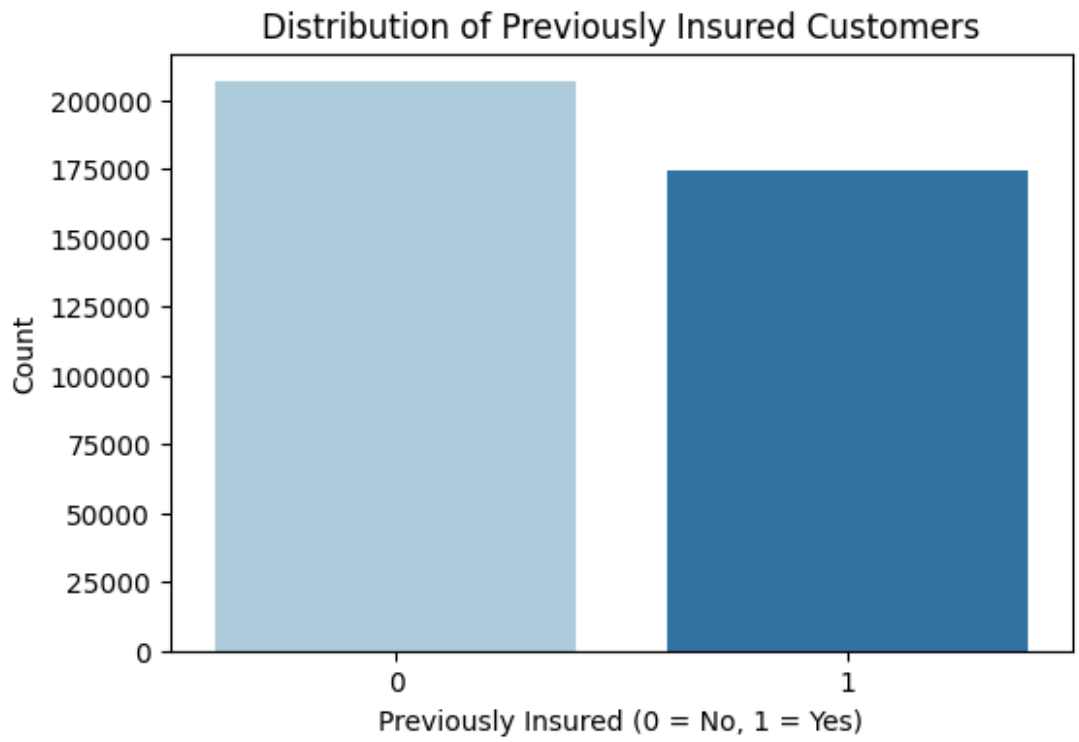



✅ ***INSIGHTS FOR AGE DISTRIBUTION:***

1. It insures that customers between age 20 to 30 are more likely to do insurance of vehicles.
1. Very few customers are under 20 or over 60, indicating that middle-aged individuals dominate the insurance-seeking population.

```
# Plot 3: Previously Insured distribution
plt.figure(figsize=(6, 4))
sns.countplot(x="Previously_Insured", data=data, palette='Paired')
plt.title('Distribution of Previously Insured Customers')
plt.xlabel('Previously Insured (0 = No, 1 = Yes)')
plt.ylabel('Count')
plt.show()
```

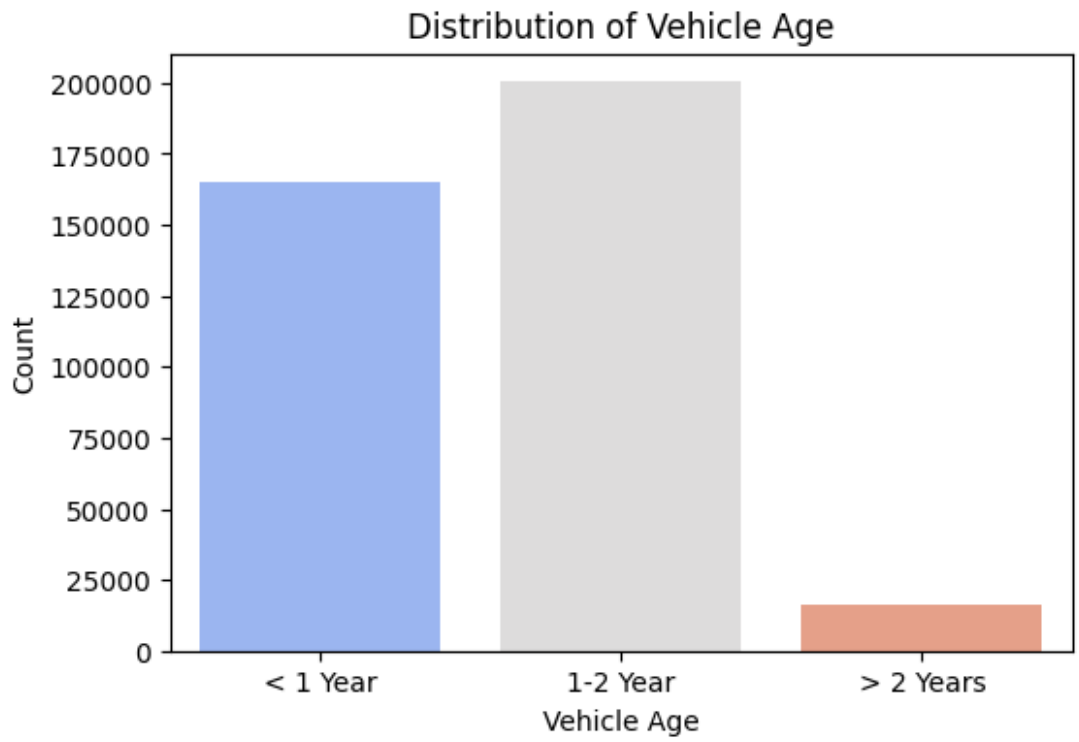
```
/tmp/ipython-input-31-3145059337.py:3: FutureWarning:

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="Previously_Insured", data=data, palette='Paired')
```




Many customers are new to insurance, implying growth potential and the need for customer education or onboarding strategies.

```
# Plot 4: Vehicle Age Distribution
plt.figure(figsize=(6, 4))
sns.countplot(x='Vehicle_Age', data=data, order=['< 1 Year', '1-2 Year', '> 2 Years'], palette='coolwarm')
plt.title('Distribution of Vehicle Age')
plt.xlabel('Vehicle Age')
plt.ylabel('Count')
plt.show()
```

```
/tmp/ipython-input-32-1551934408.py:3: 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='Vehicle_Age', data=data, order=['< 1 Year', '1-2 Year', '> 2 Years'], palette='coolwarm')
```




✅ ***INSIGHTS For Distribution of Vehicle Age –***

1. The 1-2 Year category has the highest count. This suggests a significant number of policyholders own vehicles between 1- 2 year.

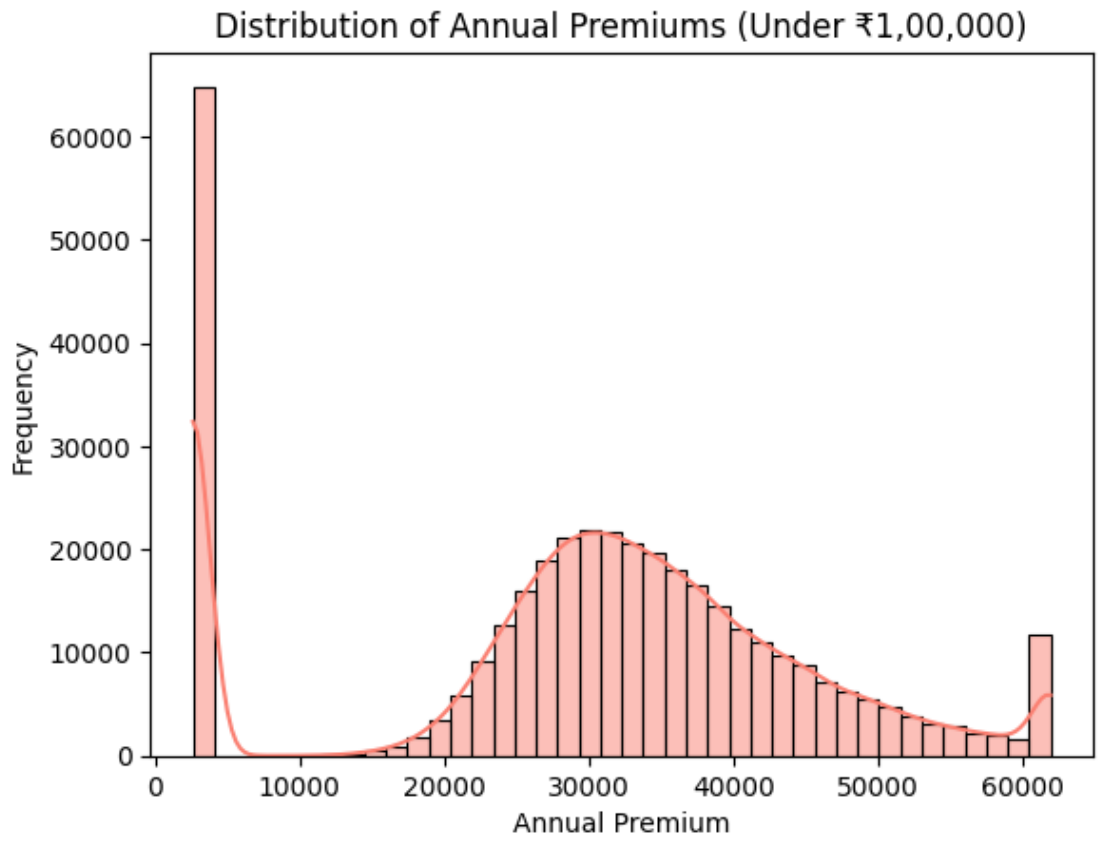
2. Owners of newer vehicles (< 1 Year) are less likely to be in the dataset.

**Reasons could include**: First-time buyers may delay personal insurance.

1. Older vehicles are seen as more vulnerable, so customers proactively seek insurance.

2. Year category is the least represented.This may be because : Target first-time buyers with attractive early insurance offers and Partner with vehicle dealers for bundled insurance.

```
# Plot 5: Annual Premium Distribution
plt.figure()
sns.histplot(data[data['Annual_Premium'] < 100000]['Annual_Premium'],
bins=40, kde=True, color='salmon')
plt.title('Distribution of Annual Premiums (Under ₹1,00,000)')
plt.xlabel('Annual Premium')
plt.ylabel('Frequency')
plt.show()
```



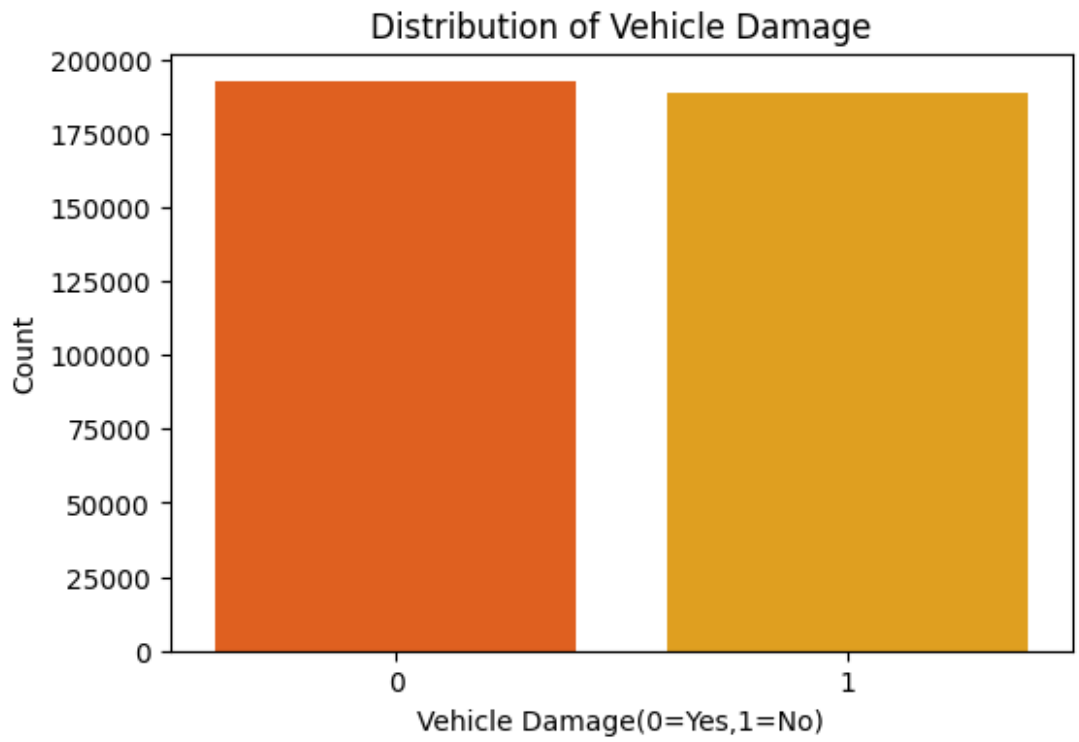

INSIGHT:

1. A large concentration of policies falls between approximately ₹20,000 to ₹45,000.

2. The KDE (Kernel Density Estimate) overlay helps visualize the probability density, reinforcing the peak around ₹30,000–₹40,000 and the long tail.

```
# Plot 6: Vehicle Damage distribution
plt.figure(figsize=(6, 4))
sns.countplot(x='Vehicle_Damage', data=data, palette='autumn')
plt.title('Distribution of Vehicle Damage')
plt.xlabel('Vehicle Damage(0=Yes,1=No)')
plt.ylabel('Count')
plt.show()
```

```
/tmp/ipython-input-34-2120160295.py:3: 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='Vehicle_Damage', data=data, palette='autumn')
```

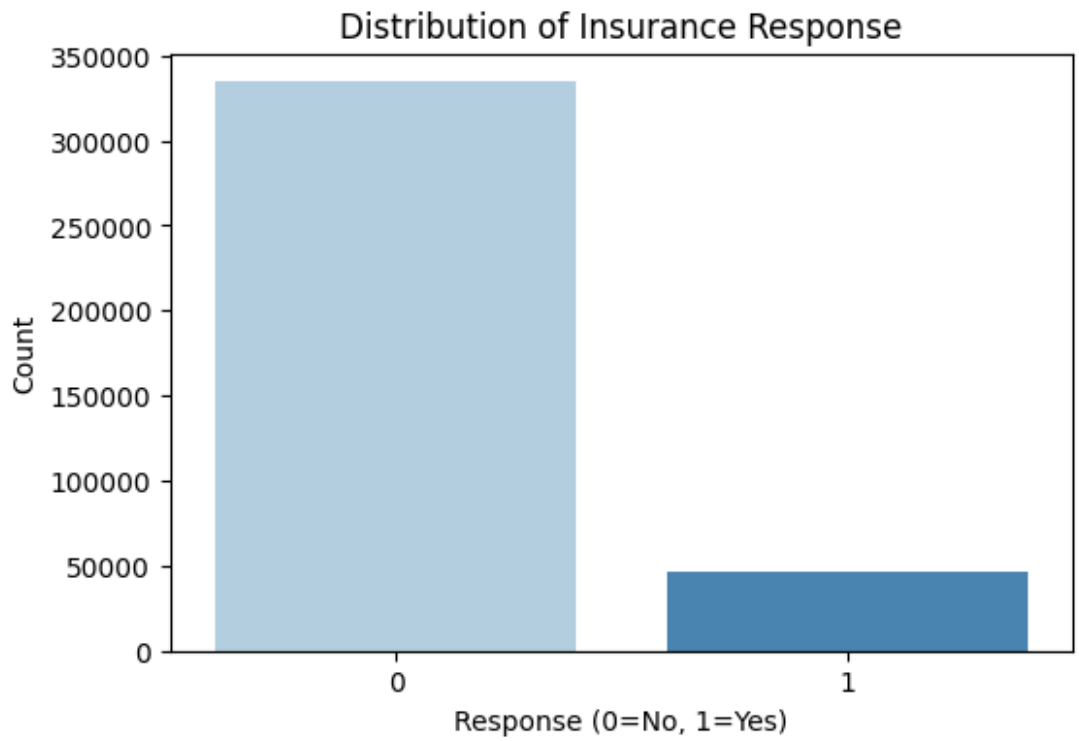



✅ ***INSIGHTS:*** The Vehicle_Damage column has two categories: 'Yes' (damage reported) and 'No' (no damage).

The plot likely shows more customers with vehicle damage, indicating that most vehicles in the dataset are undamaged as compared to vehicles that are not damaged.

```
# Plot 7: Response variable distribution (Target)
plt.figure(figsize=(6, 4))
sns.countplot(x='Response', data=data, palette='Blues')
plt.title('Distribution of Insurance Response')
plt.xlabel('Response (0=No, 1=Yes)')
plt.ylabel('Count')
plt.show()
```

```
/tmp/ipython-input-35-2467555102.py:3: 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='Blues')
```




✅ ***INSIGHTS:***

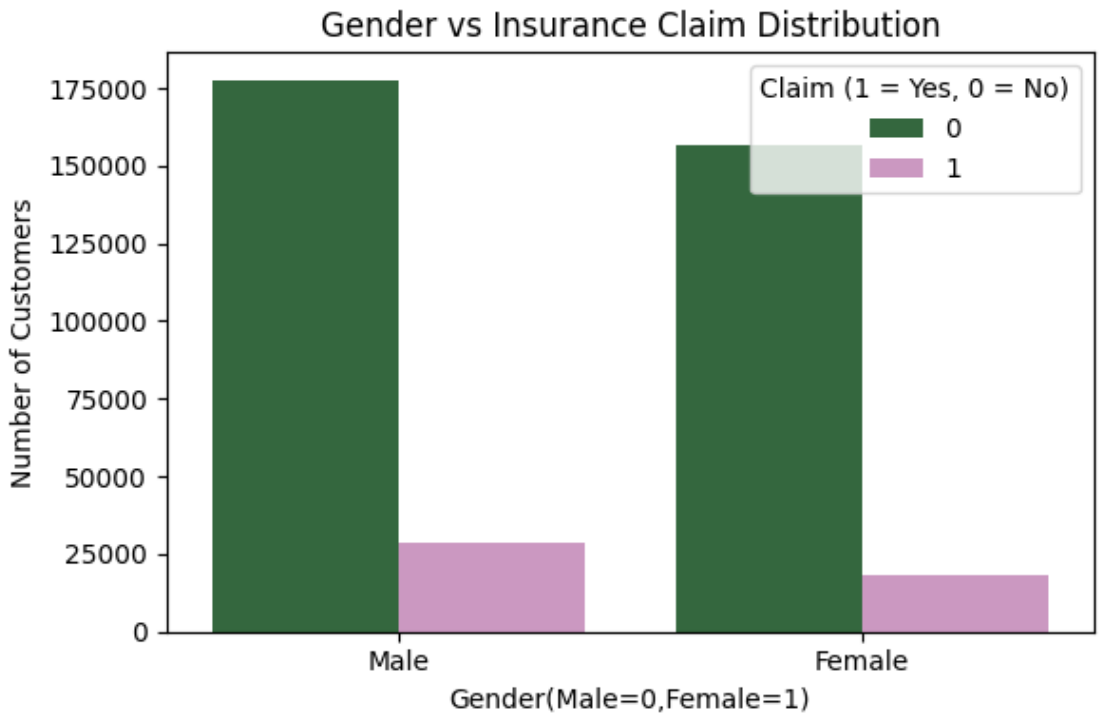
1. The plot likely shows a much higher number of 0s than 1s, indicating most customers did not purchase insurance.
2. The proportion of customers who accepted the insurance offer (Response = 1) is relatively low.

## ***Step 4.Feature Analysis:***

Examine the relationship between features and the target variable (insurance claims).

```
data['Gender'] = data['Gender'].replace({0: 'Male', 1: 'Female'})
```

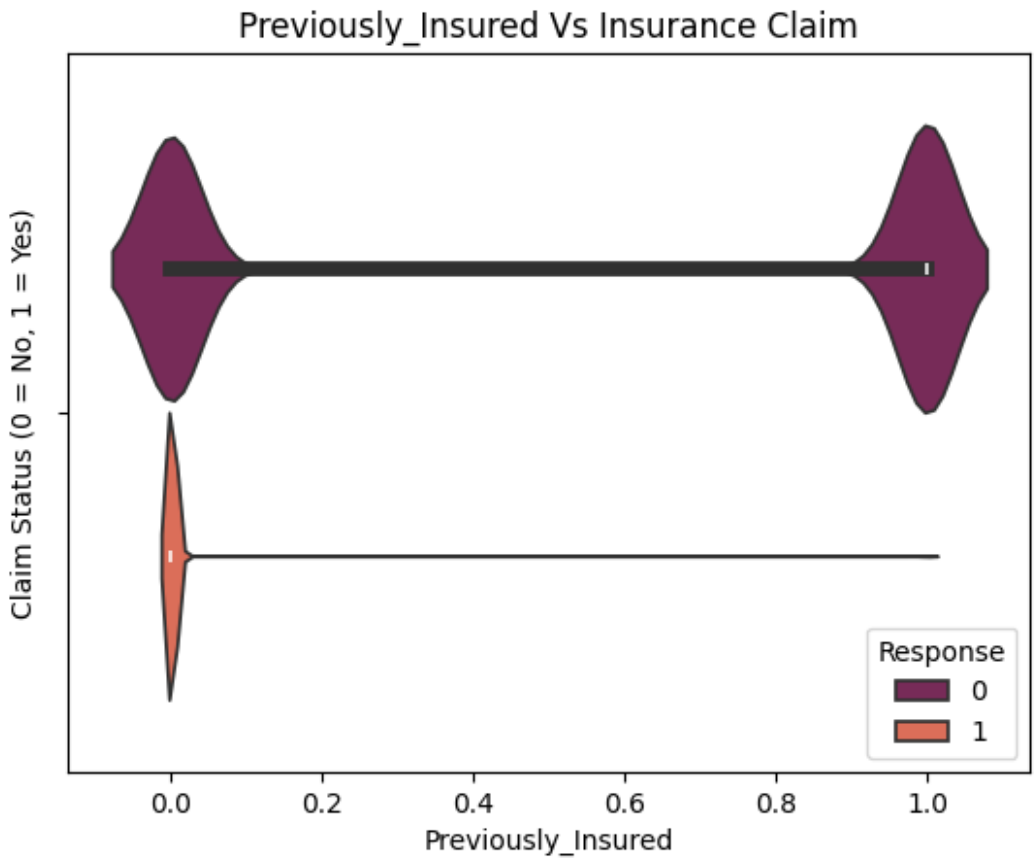
```
#Plot 1. Countplot of Gender Vs Insurance Claim
plt.figure(figsize=(6, 4))
sns.countplot(data=data, x='Gender', hue='Response', palette='cubehelix')
plt.title('Gender vs Insurance Claim Distribution')
plt.xlabel('Gender(Male=0,Female=1)')
plt.ylabel('Number of Customers')
plt.legend(title='Claim (1 = Yes, 0 = No)')
plt.tight_layout()
plt.show()
```




## ***Key Insights***

1. The male bar (x = Male) signifies that the dataset is male-dominated than females.
2. The claim rates Appear Similar Across Genders as the proportion of claims (Response = 1) within male and female bars is visually similar.

```
#Plot 2. Violinplot plot of Previously Insured Vs Insurance Claim
sns.violinplot(x='Previously_Insured',hue="Response",data=data,palette="rocket")
plt.title("Previously_Insured Vs Insurance Claim")
plt. xlabel("Previously_Insured")
plt. ylabel("Claim Status (0 = No, 1 = Yes)")
plt.show()
```

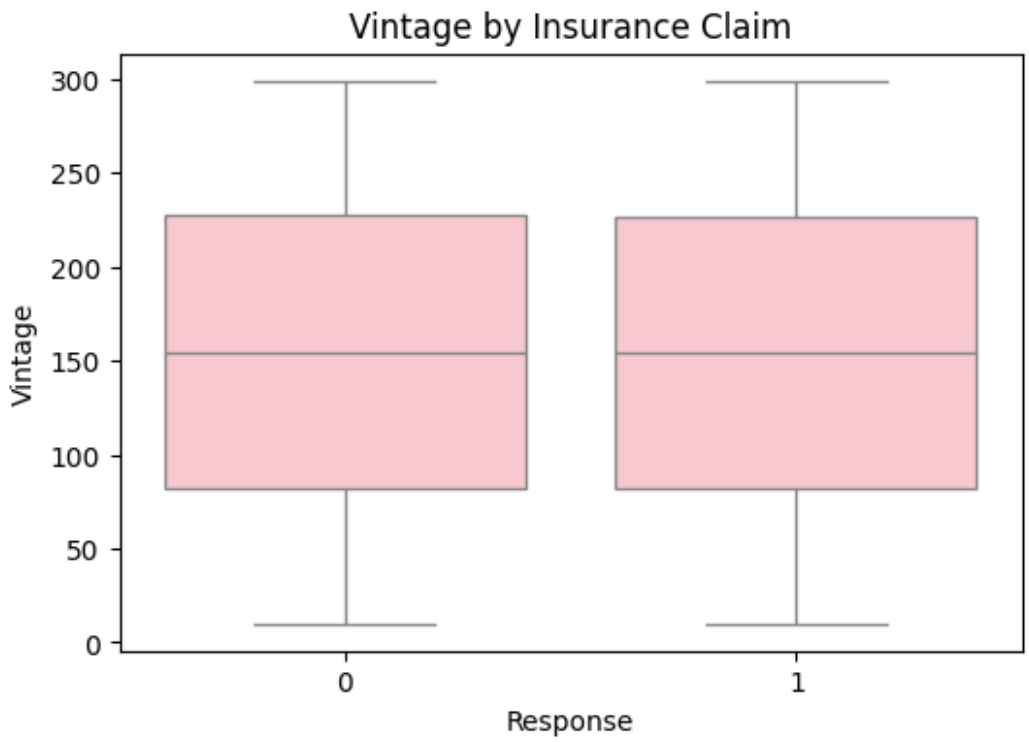



🧠 ***Insights from Previously_Insured vs Response:***

✅ 1. Most of the Claims Come from Uninsured Individuals as Customers who were not previously insured are much more likely to file a claim now.

✅ 2. Previously Insured Customers Have Low Claim Rates If the violin for Previously_Insured = 1 is concentrated then it means that people already insured tend to not file claims—possibly because they are lower-risk or already have other coverage.

```
# Plot 3. Boxplot of Vintage by Insurance Claim
feature="Vintage"
plt.figure(figsize=(6,4))
sns.boxplot(data=data, x='Response', y=feature,color="pink")
plt.title(f"{feature} by Insurance Claim")
plt.show()
```



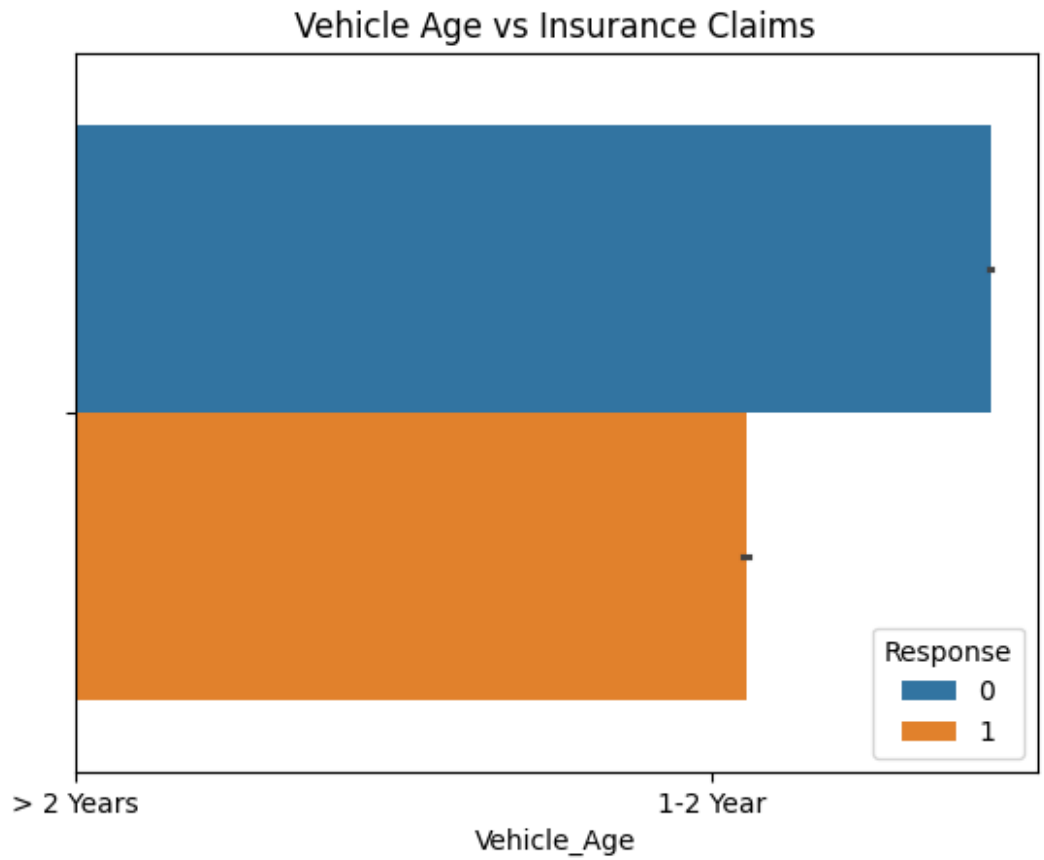

✅ ***Interpretation of Boxplot:***

Insurance_Claim for Vintage is almost same.Only a litte diffence can be seen .

📌 1. The median vintage for Response = 1 (claimants) is lower than for Response = 0, then it means that customers who recently joined the company are more likely to file claims.

📌 2. Higher Vintage = More Stable,if Response = 0 has a wider box with a higher median, then it means that long-standing customers are less likely to file claims. It may be due to loyalty, trust, or already being educated about policy terms

```
#Plot4.  Barplot of Vehicle Age vs Insurance Claims
sns.barplot(x='Vehicle_Age', hue='Response', data=data,)
plt.title('Vehicle Age vs Insurance Claims')
plt.xlabel("Vehicle_Age")
plt.show()
```




## ***Insights from the Plot***

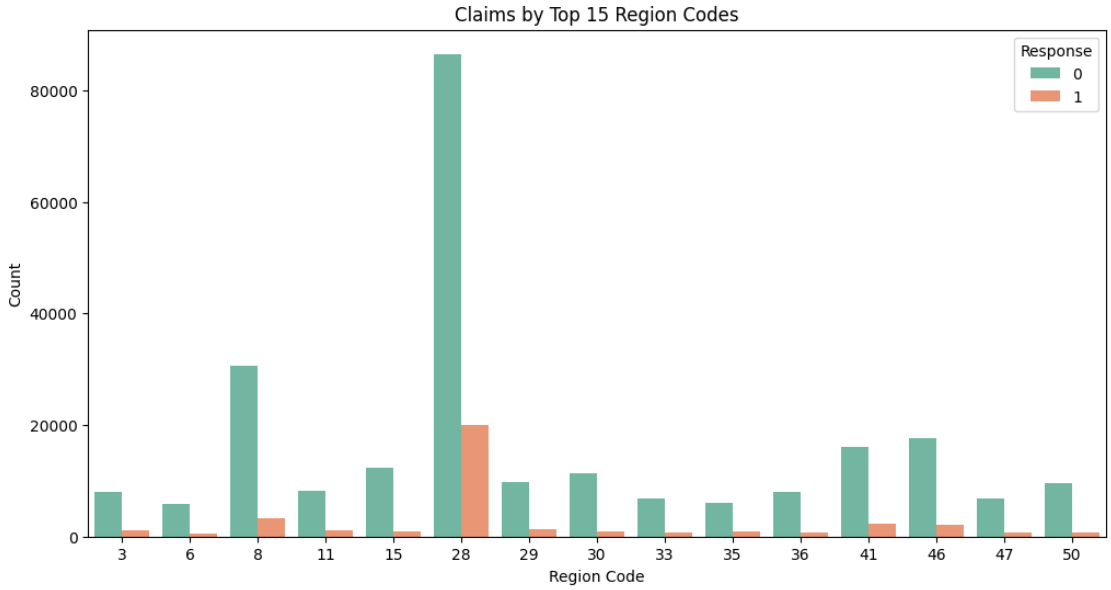
1. Customers with older vehicles (especially >2 years old) are more likely to file claims.

This could be due to: More wear and tear, More repair-prone, Lower vehicle value → more claims for minor damage

1. New vehicles are less likely to be involved in claims, possibly due to:

Better safety tech,more cautious driving,due to warranties given by the salers.

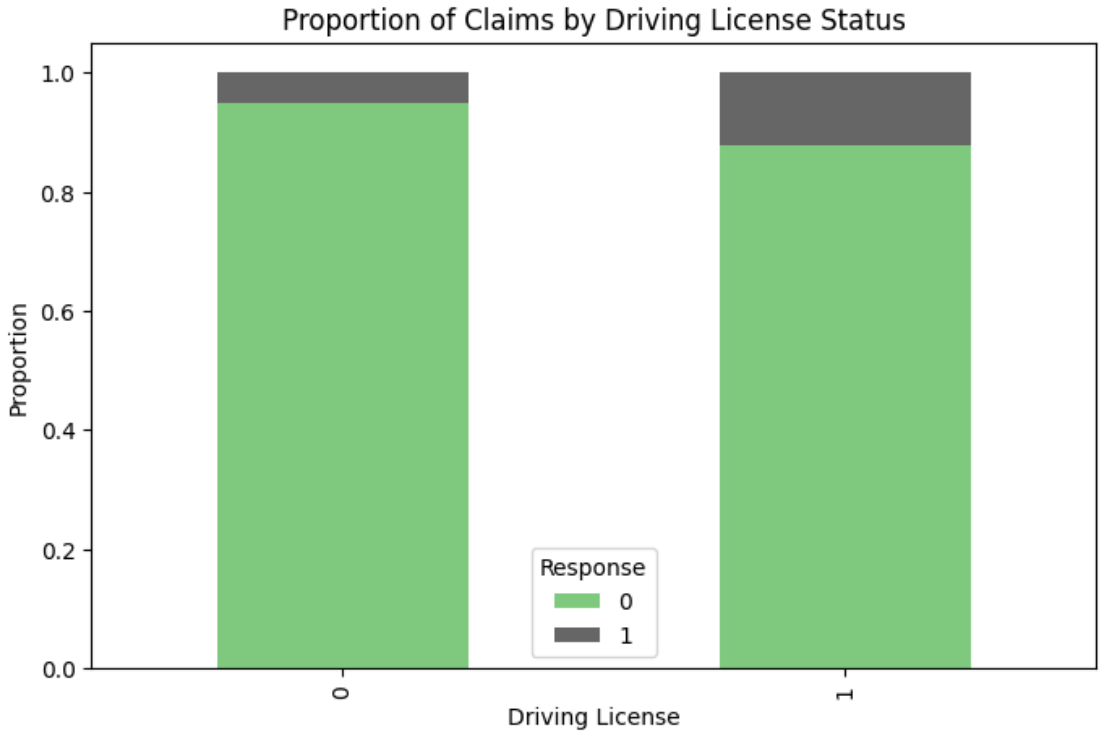
```
#Plot 5. Countplot of claims by top 15 Regions
top_regions = data['Region_Code'].value_counts().nlargest(15).index
filtered_df = data[data['Region_Code'].isin(top_regions)]

plt.figure(figsize=(12, 6))
sns.countplot(data=filtered_df, x='Region_Code', hue='Response',
palette='Set2')
plt.title('Claims by Top 15 Region Codes')
plt.xlabel('Region Code')
plt.ylabel('Count')
plt.legend(title='Response')
plt.show()
```




## 📊 ***Insights from the Plot:***

Some regions which are taller have no claims while there are also some regions which have fewer claims.

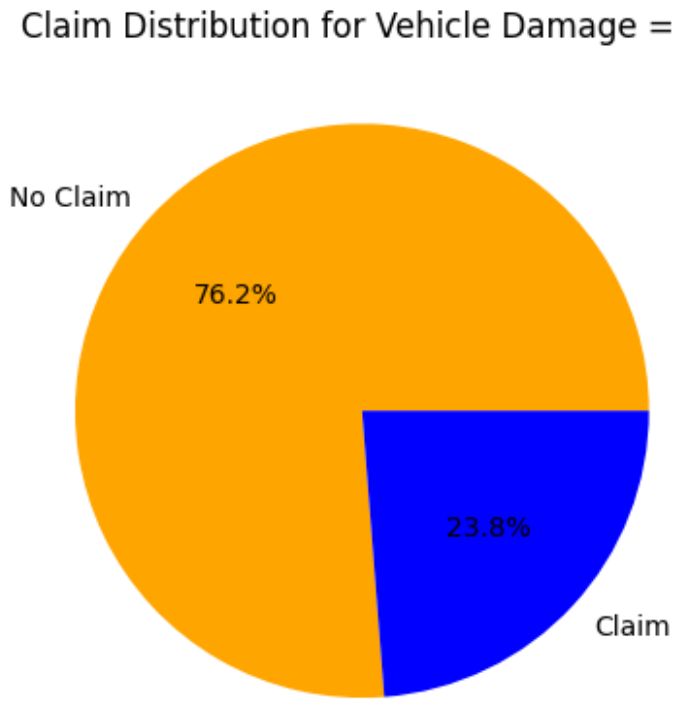
```
# Plot 6. Proportion of Claims by Driving License Status
license_claims = pd.crosstab(data['Driving_License'], data['Response'],
normalize='index')
license_claims.plot(kind='bar', stacked=True, colormap='Accent',
figsize=(8,5))
plt.title('Proportion of Claims by Driving License Status')
plt.xlabel('Driving License')
plt.ylabel('Proportion')
plt.legend(title='Response')
plt.show()
```

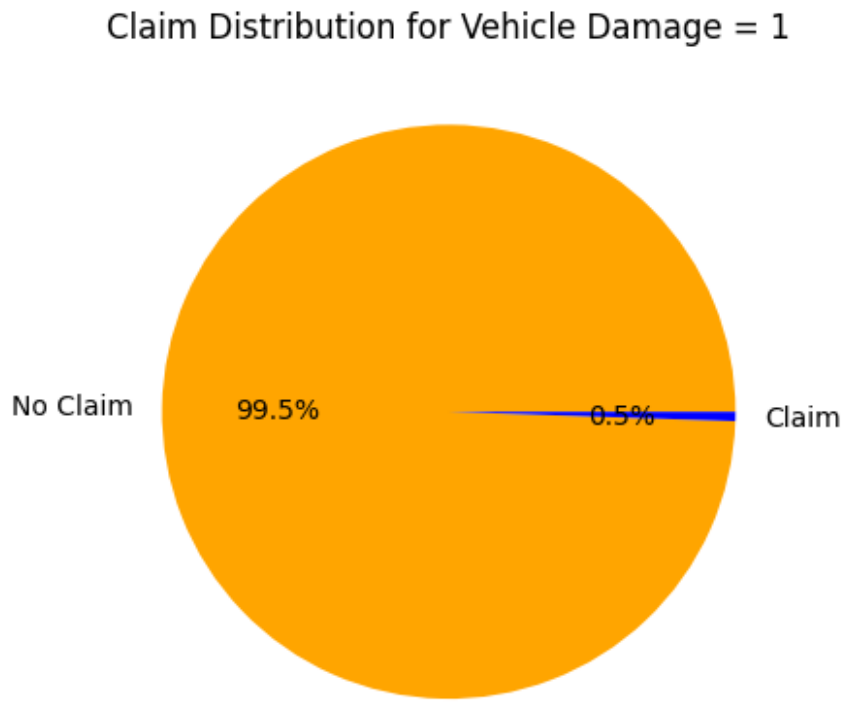



✅ ***Insights From The Plot:***

1. Most license holders don't make claims, but there's still a noticeable share that does.
1. Individuals without a license are much less likely to make a claim, either due to not actively driving or being filtered out in policy approval.

```
#Plot 7.  Pieplot for Claim Distribution for Vehicle Damage
unique_damage_values = data["Vehicle_Damage"].unique()

for damage in unique_damage_values:
  labels = ["No Claim", "Claim"]
  sizes = data[data["Vehicle_Damage"] ==
damage]["Response"].value_counts().sort_index()

  plt.pie(sizes, labels=labels, colors=["orange", "blue"], autopct='%1.1f%%')
  plt.title(f"Claim Distribution for Vehicle Damage = {damage}")
  plt.show()
```



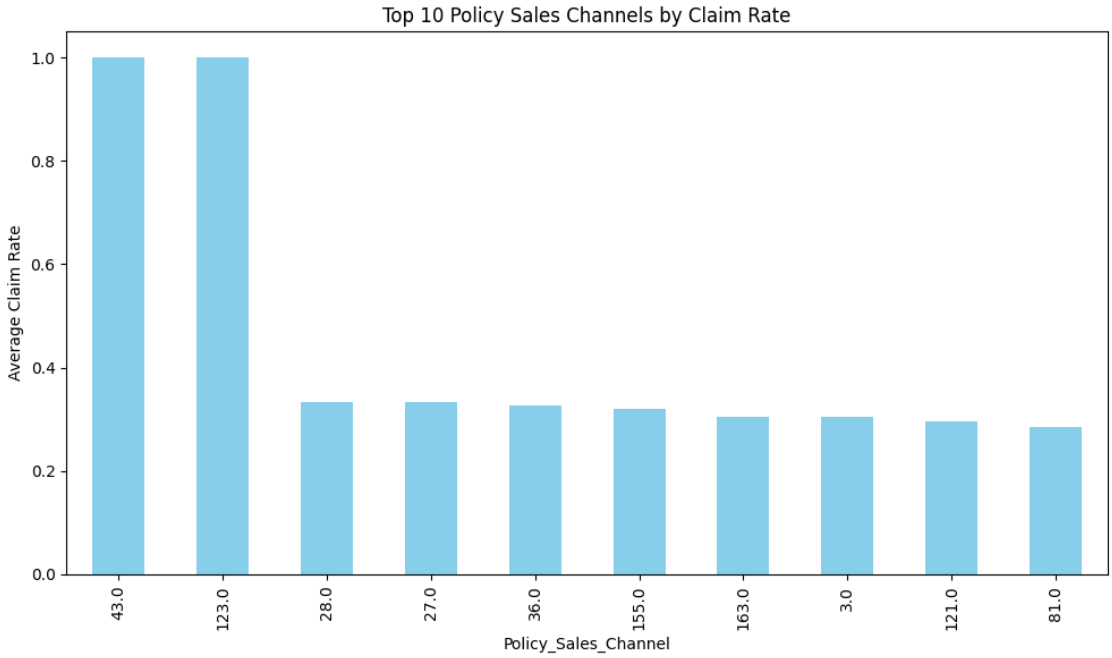

✅ ***Insights from pie plot:***

1. Individuals with prior vehicle damage are more likely to file an insurance claim. This makes intuitive sense, as damaged vehicles are riskier and more likely to need repairs or replacements.

2. Customers with undamaged vehicles are less likely to make a claim, indicating they may be safer drivers or own newer/maintained cars.

***#Plot 8. Top 10 Policy Sales Channels by Claim Rate***

```
claim_rate =
data.groupby('Policy_Sales_Channel')['Response'].mean().sort_values(ascending
=False)
top_channels = claim_rate.head(10)
top_channels.plot(kind='bar', figsize=(10,6), color='skyblue')
plt.ylabel('Average Claim Rate')
plt.title('Top 10 Policy Sales Channels by Claim Rate')
plt.tight_layout()
plt.show()
```



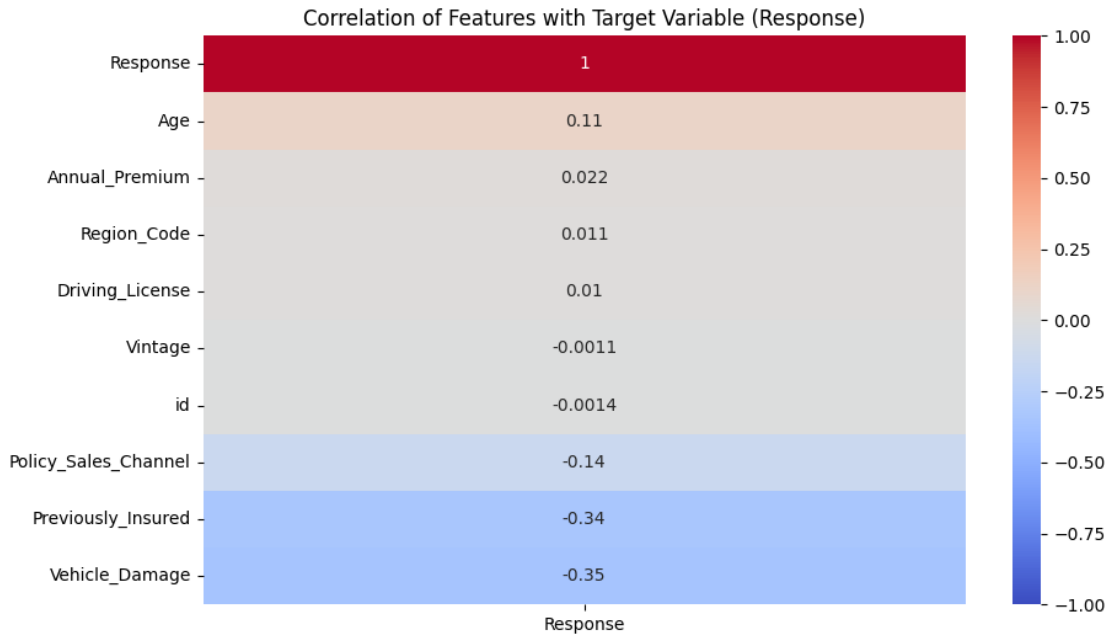

✅ ***Insights from the plot:***

A high claim rate doesn't always mean the channel is bad — it could mean: The channel is reaching high-risk but underserved customers. There's misalignment between product and customer risk profile.

***#plot 9. Heatmap between target and features.***

```
numerical_data = data.select_dtypes(include=['int64', 'float64'])
dff = numerical_data.corr()

plt.figure(figsize=(10,6))
sns.heatmap(dff[['Response']].sort_values(by='Response', ascending=False),
```

```
            annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title("Correlation of Features with Target Variable (Response)")
plt.show()
```




## ✅ ***Insights from heapmap:***

1. ✅ Positive Correlation (Closer to +1):

These features increase as the probability of making a claim increases:

E.g., Vehicle_Age, Annual_Premium may be more likely to file insurance claim. Customers paying higher premiums might be more likely to file claims, possibly due to having costlier vehicles or comprehensive coverage.

2. ❌ Negative Correlation (Closer to -1):

These features decrease as the probability of making a claim increases:

E.g., Previously insured individuals may be less likely to make a claim, possibly because they maintain their vehicles better or are lower risk.

If age has a negative correlation, younger individuals may have a higher likelihood of claims.

3. 🔁 Near Zero Correlation (Close to 0):

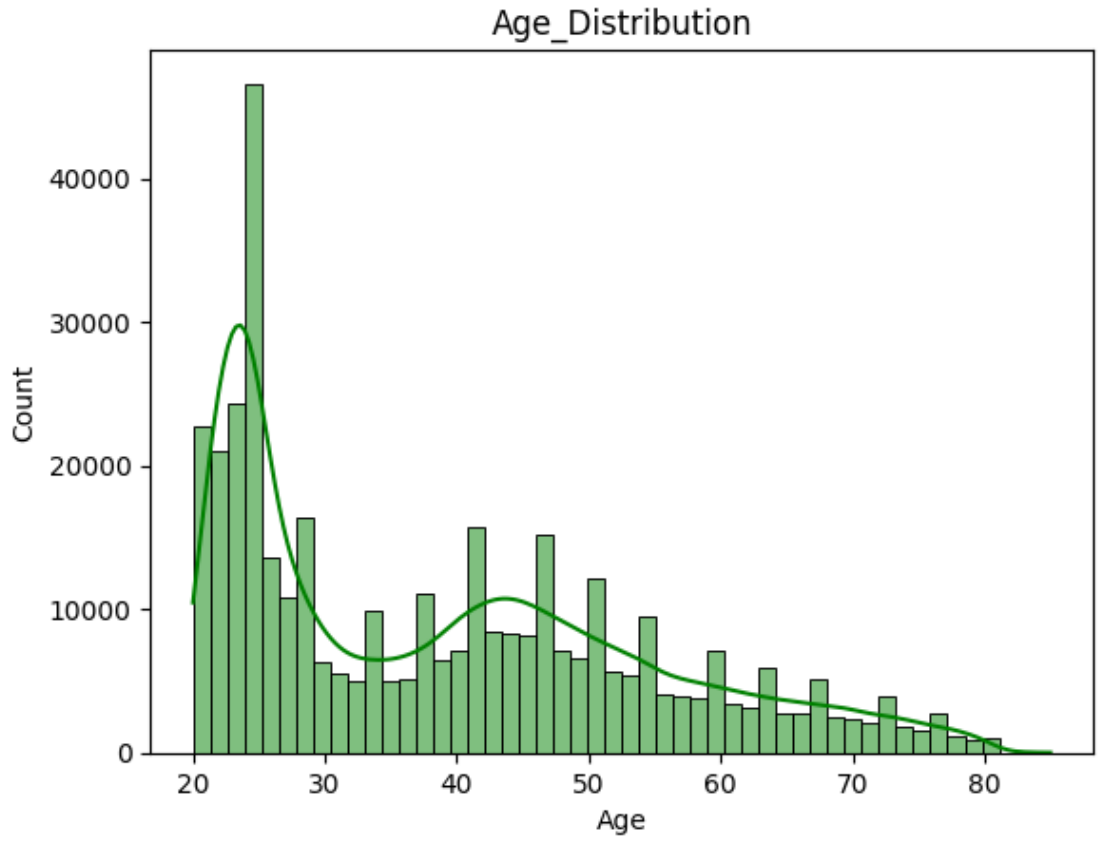
These features have little to no linear relationship with the target:

E.g., Policy_Sales_Channel, Vintage, etc., may not have a strong influence on the claim response in terms of correlation.

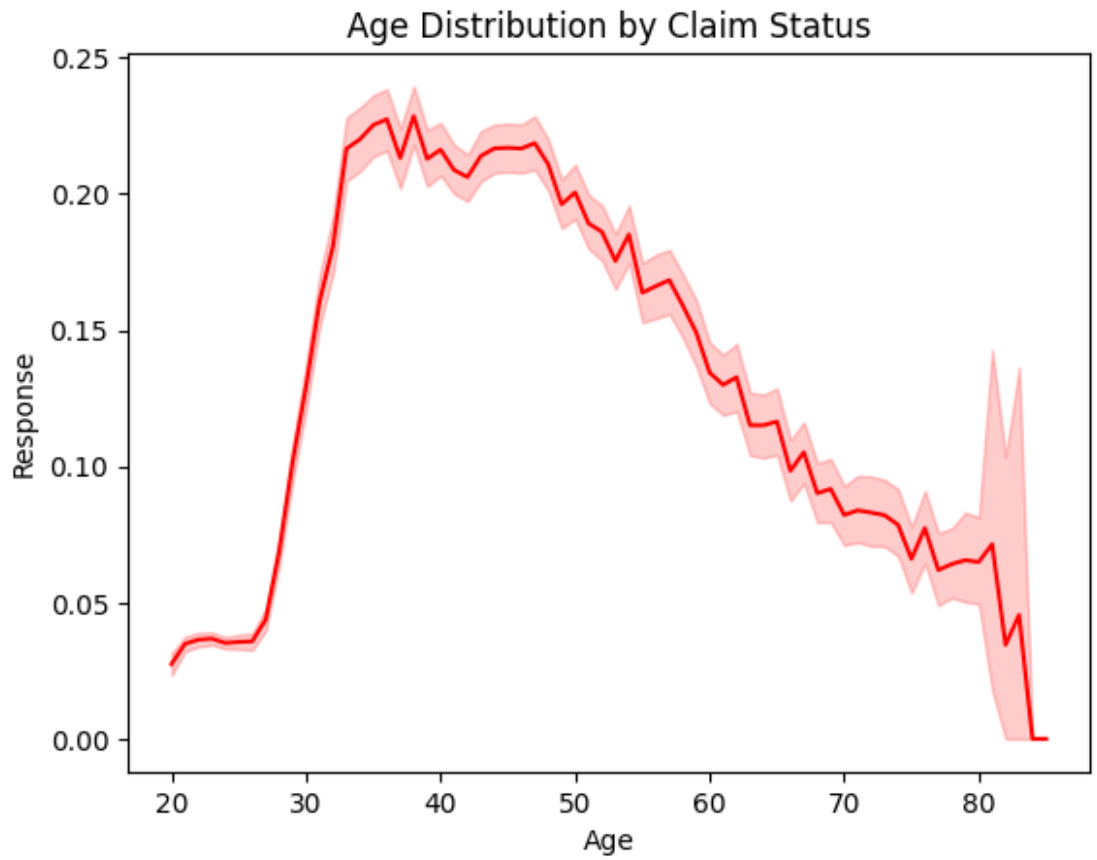
## *Step 5. Age Distribution*

Analyze the age distribution within the dataset and its impact on insurance claims

```
#Histplot to analyse the distribution of age.
sns.histplot(data["Age"],bins=50,kde=True,color="green")
plt.title("Age_Distribution")
plt.show()
```

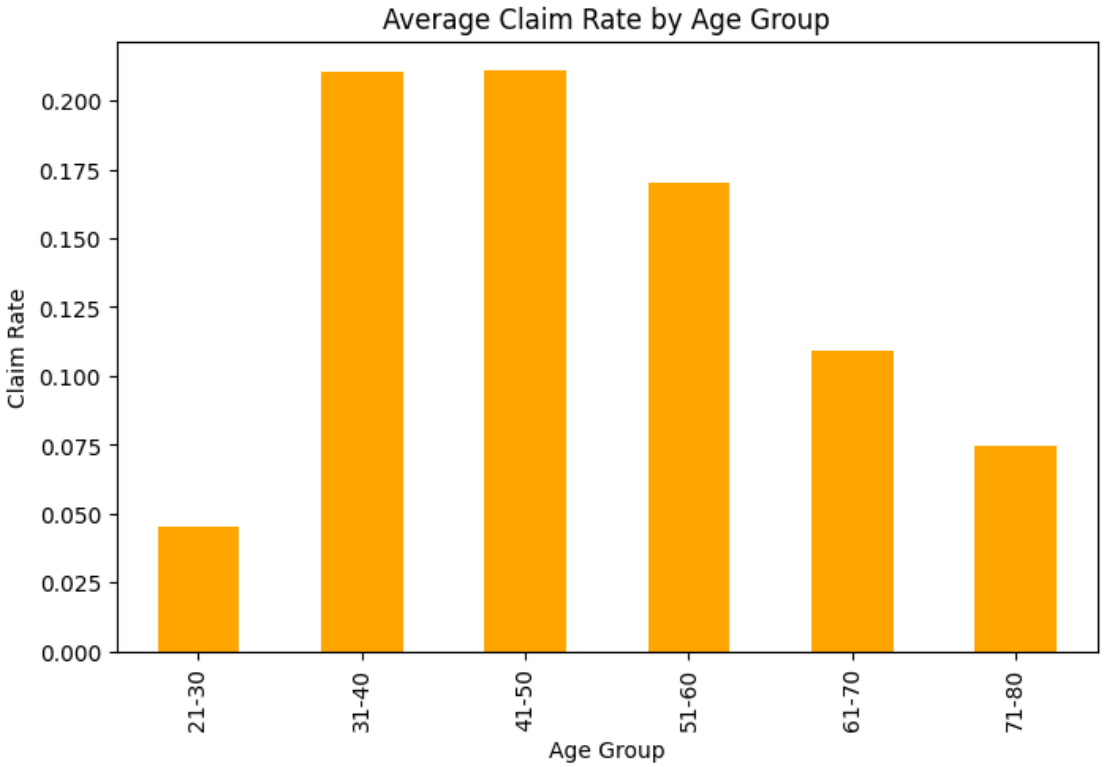



```
# Lineplot to show the impact of insurance claim w.r.t Age.
sns.lineplot(x="Age", y="Response",data=data,color="red")
plt.title('Age Distribution by Claim Status')
plt.xlabel("Age")
plt.ylabel("Response")
plt.show()
```




```
# Average Claim Rate by Age Group
data['Age_Group'] = pd.cut(data['Age'], bins=[20,30,40,50,60,70,80],
                           labels=['21-30','31-40','41-50','51-60','61-70','71-
80'])

group_claims = data.groupby('Age_Group')['Response'].mean()

group_claims.plot(kind='bar', color='orange', figsize=(8,5))
plt.title('Average Claim Rate by Age Group')
plt.ylabel('Claim Rate')
plt.xlabel('Age Group')
plt.show()
```

```
/tmp/ipython-input-48-4106135174.py:5: 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.
  group_claims = data.groupby('Age_Group')['Response'].mean()
```



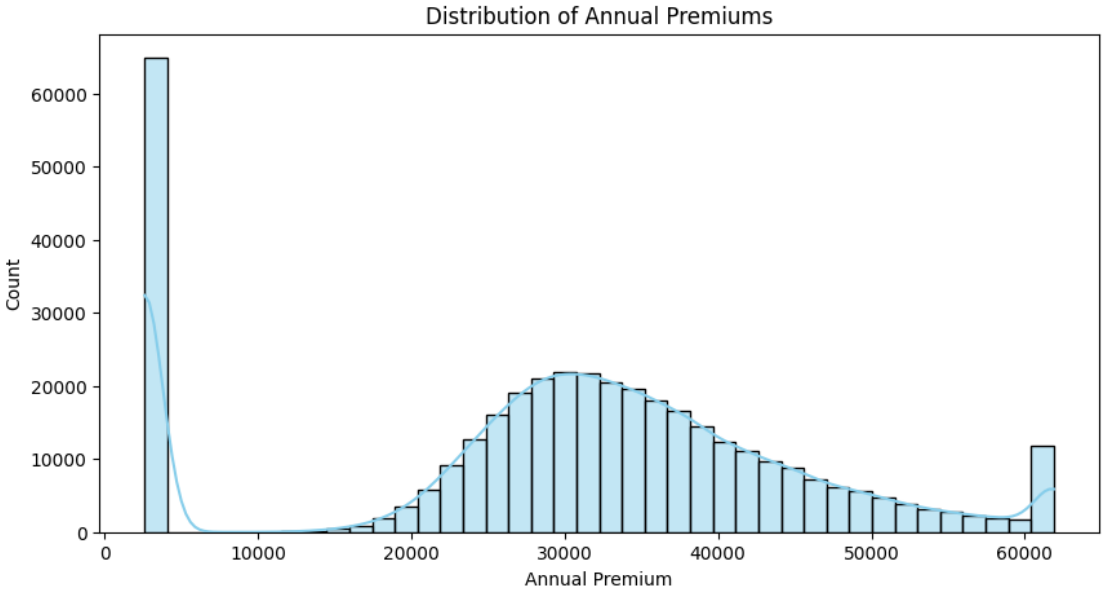

## ***Step 6.Premium Analysis***

Investigate the distribution of insurance premiums and their correlation with claim frequencies.

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

```
count    381109.000000
mean      30148.169353
std       15476.398118
min        2630.000000
25%       24405.000000
50%       31669.000000
75%       39400.000000
max       61892.500000
Name: Annual_Premium, dtype: float64
```

```
# Distribution of Annual Premium
plt.figure(figsize=(10, 5))
sns.histplot(data['Annual_Premium'], kde=True,bins=40, color='skyblue')
plt.title('Distribution of Annual Premiums')
plt.xlabel('Annual Premium')
plt.ylabel('Count')
plt.show()
```




```
dff=data[["Annual_Premium","Response"]]
dff
```

| | Annual_Premium | Response |
|---|---|---|
| **0** | 40454.0 | 1 |
| **1** | 33536.0 | 0 |
| **2** | 38294.0 | 1 |
| **3** | 28619.0 | 0 |
| **4** | 27496.0 | 0 |
| **...** | ... | ... |
| **381104** | 30170.0 | 0 |
| **381105** | 40016.0 | 0 |
| **381106** | 35118.0 | 0 |
| **381107** | 44617.0 | 0 |
| **381108** | 41777.0 | 0 |

381109 rows × 2 columns

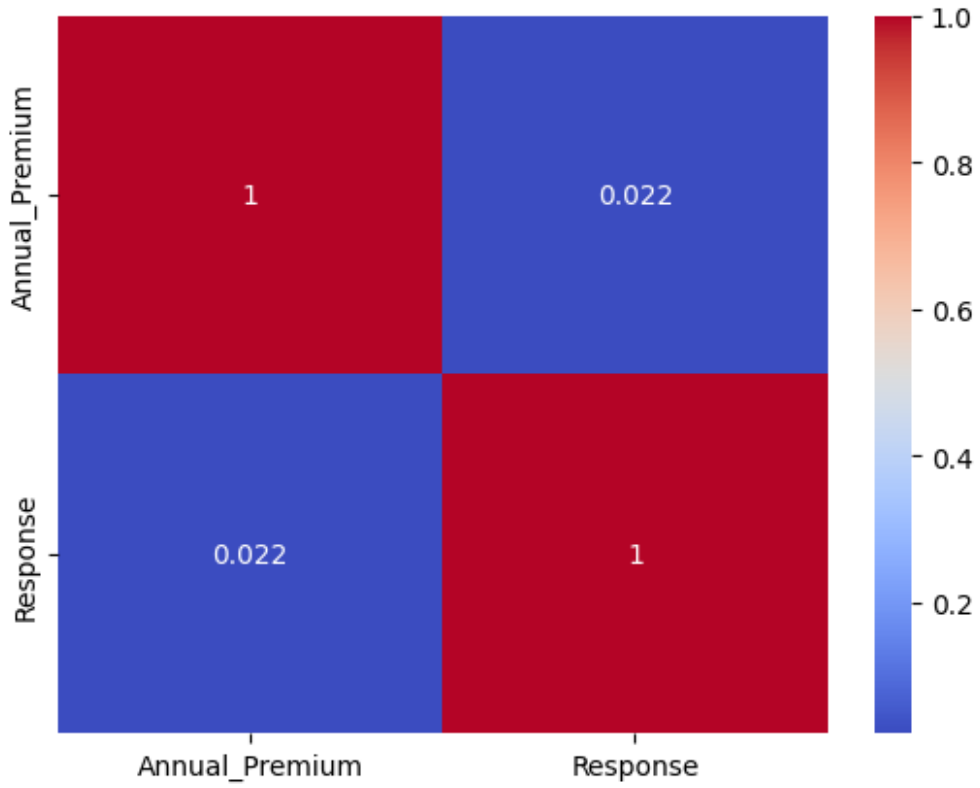
{"type":"dataframe","variable_name":"dff"}

```
dff.corr()
```

| | Annual_Premium | Response |
|---|---|---|
| **Annual_Premium** | 1.000000 | 0.022054 |
| **Response** | 0.022054 | 1.000000 |

```
#Correlation between Annual Premium and claim Frequencies.
sns.heatmap(dff.corr(),annot=True,cmap="coolwarm")
```

```
<Axes: >
```




### 🧠 *Key Insights:*

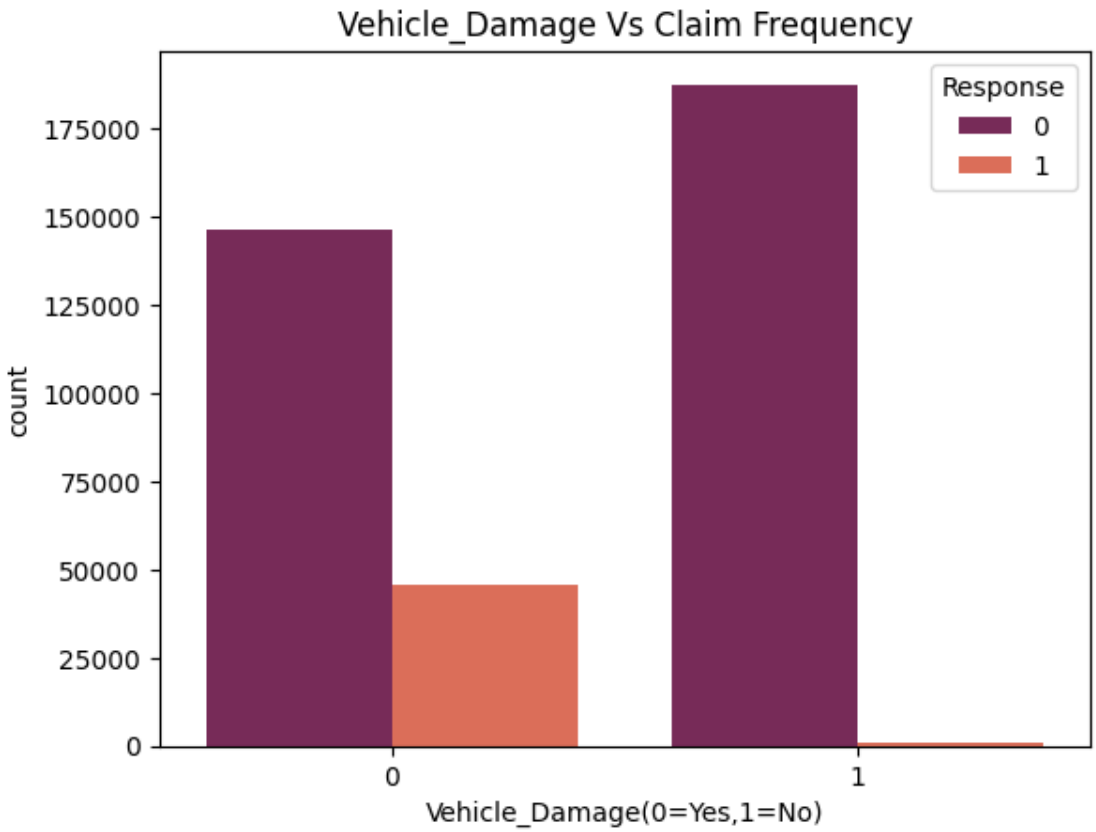
A positive correlation might suggest higher premiums are linked with higher claims.

If no correlation is found, it might indicate that premium values don't influence claim behavior.

## *Step 7. Claim Frequencies:*

Explore factors contributing to higher claim frequencies.

```
#Plot 1. Countplot of Vehicle Damage Vs Claim Frequency
sns.countplot(x='Vehicle_Damage', hue='Response', data=data,
palette='rocket')
plt.title(" Vehicle_Damage Vs Claim Frequency")
plt.xlabel("Vehicle_Damage(0=Yes,1=No)")
plt.ylabel("count")
plt.show()
```

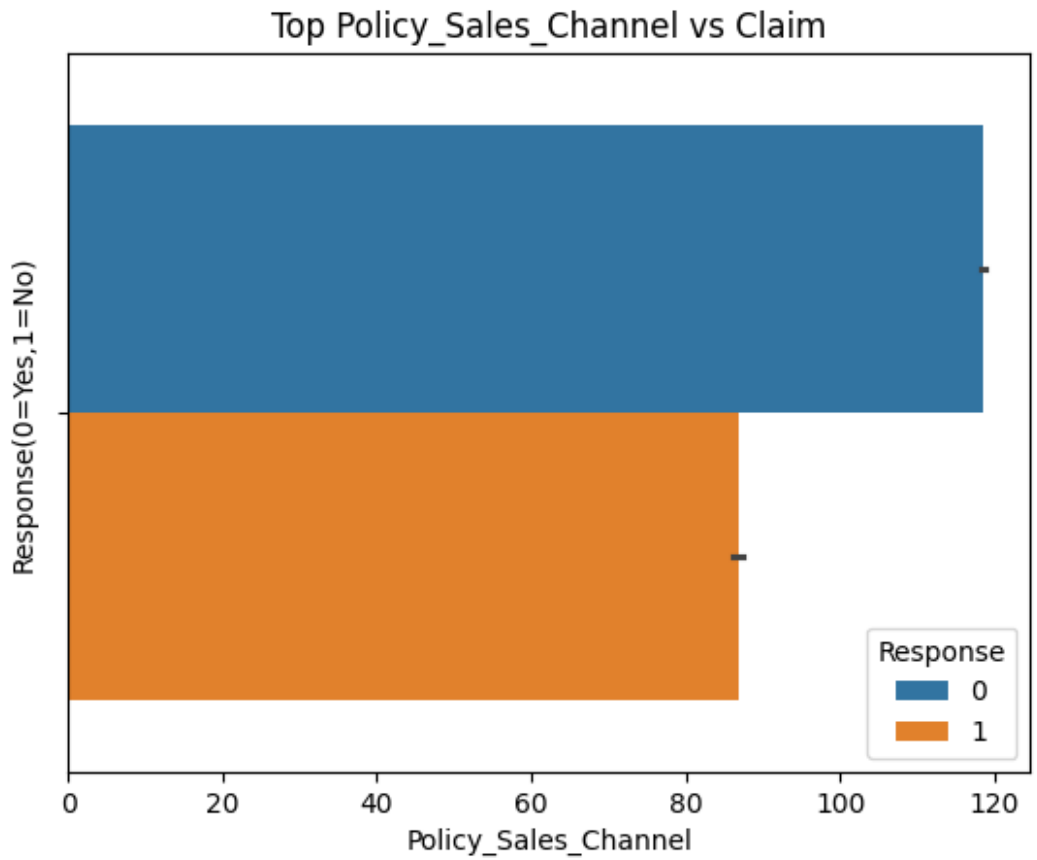



🧠 ***Interpretation of the Plot:***

For Vehicle_Damage = Yes, the count of Response = 1 (interested in policy / claim) is significantly higher than for No.

For Vehicle_Damage = No, Response = 1 is usually much lower.

```
# Plot 2. Barplot of Top Policy Sales Channel vs Claim Frequency
top_channels = data['Policy_Sales_Channel'].value_counts().nlargest(5).index
sns.barplot(data=data[data['Policy_Sales_Channel'].isin(top_channels)],
x='Policy_Sales_Channel', hue='Response')
plt.title("Top Policy_Sales_Channel vs Claim")
plt.xlabel("Policy_Sales_Channel")
plt.ylabel("Response(0=Yes,1=No)")
plt.show()
```

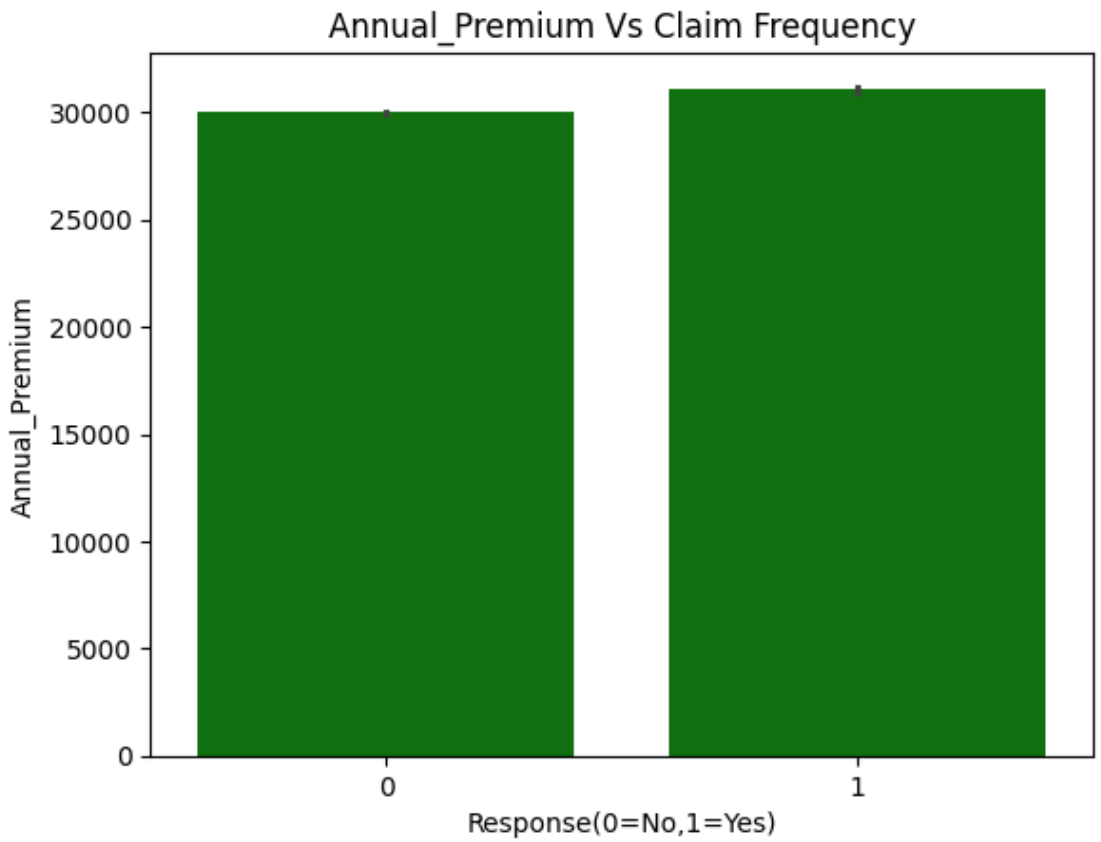



🧠 ***INSIGHTS:***

1. A tall bar in a specific channel indicates better sales or higher claim conversion.
1. A channel with mostly Response = 1may require better targeting or follow-up.

Comparing proportions helps decide which channels are most effective for customer acquisition with claim interest.

```
# Plot 3. Barplot of Annual_Premium Vs Claim Frequency
sns.barplot(x='Response', y='Annual_Premium', data=data,color="green")
plt.title("Annual_Premium Vs Claim Frequency")
plt.xlabel("Response(0=No,1=Yes)")
plt.ylabel("Annual_Premium")
plt.show()
```

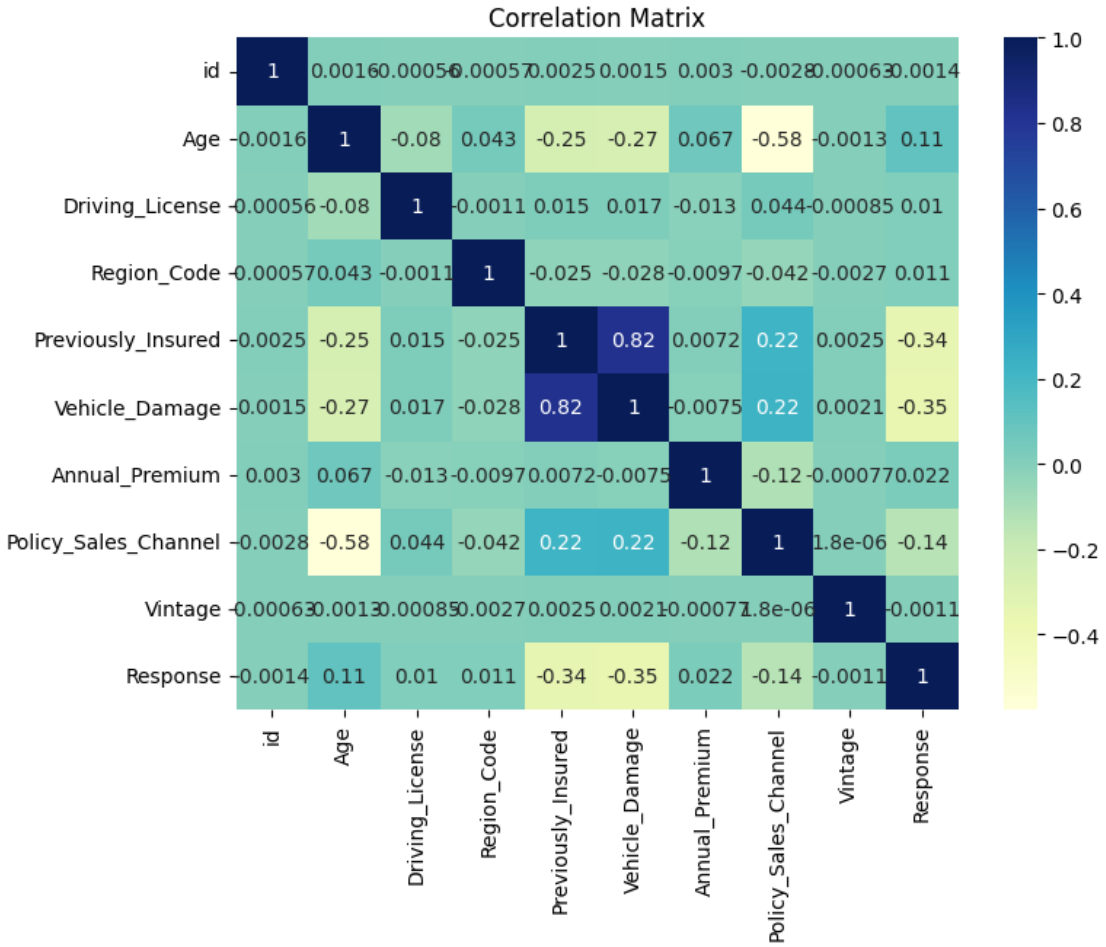



🧠 ***INSIGHTS:***

1. If Response = 1 has a higher average premium, it may suggest that people with more valuable policies are more concerned about protection.

1. If lower, it could imply price-sensitive customers are more insurance-conscious.

```
#Plot4 . Heatmap of correlation matrix
plt.figure(figsize=(8,6))
numerical_data = data.select_dtypes(include=['int64', 'float64'])
sns.heatmap(numerical_data.corr(), annot=True, cmap='YlGnBu')
plt.title("Correlation Matrix")
plt.show()
```




## 🧠 ***INSIGHTS :***

Positively correlated features (e.g., Vehicle_Damage, Vehicle_Age) might be predictors of higher claim interest.

Negatively correlated features (e.g., Previously_Insured) might be inversely related to claim interest.

## ***Step 8. Gender Analysis***

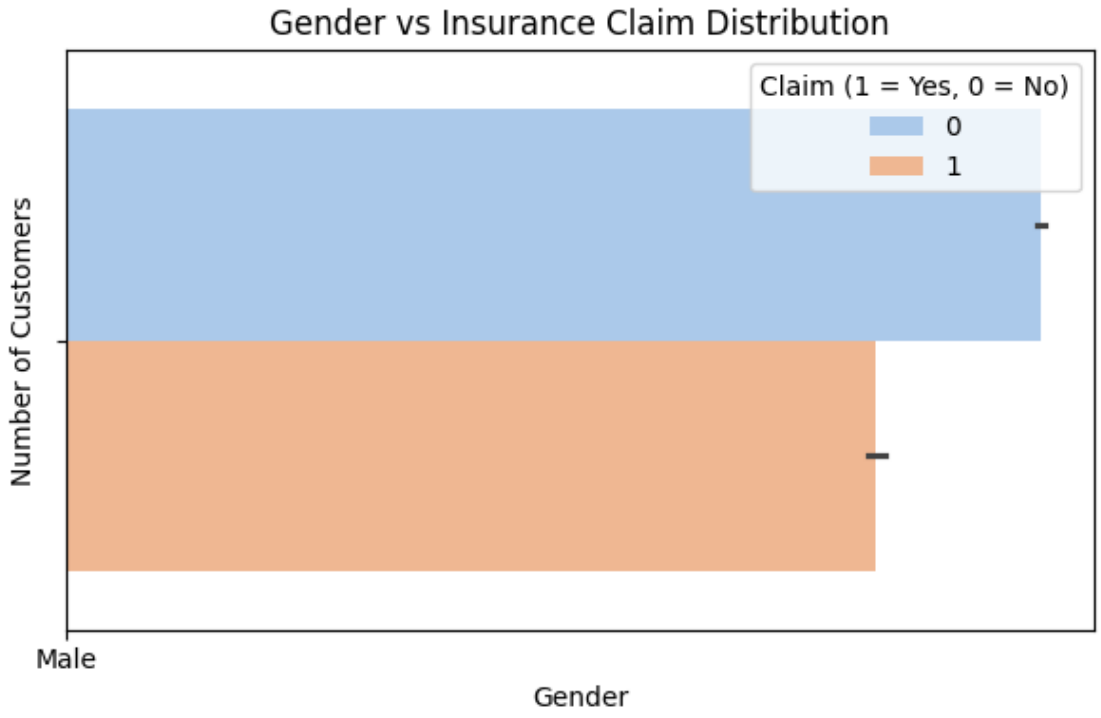
Investigate the role of gender in insurance claims.

```
data['Gender'].value_counts()
```

```
Gender
Male      206089
Female    175020
Name: count, dtype: int64
```

```
#Barplot of Gender vs Insurance Claim Distribution
plt.figure(figsize=(6, 4))
sns.barplot(data=data, x='Gender', hue='Response', palette='pastel')
```

```
plt.title('Gender vs Insurance Claim Distribution')
plt.xlabel('Gender')
plt.ylabel('Number of Customers')
plt.legend(title='Claim (1 = Yes, 0 = No)')
plt.tight_layout()
plt.show()
```




## 🔍 ***Possible Insights:***

If one gender has a significantly higher claim rate, it may indicate a behavioral or risk profile difference.

If both genders have similar claim rates, it suggests that gender may not be a strong predictor for claims.

Business application: Decide whether gender should be a feature in underwriting or risk prediction models.

## ***Step 9. Vehicle Age and Claims***

**Examine the impact of vehicle age on the likelihood of a claim**

```
data['Vehicle_Age'].value_counts()
```

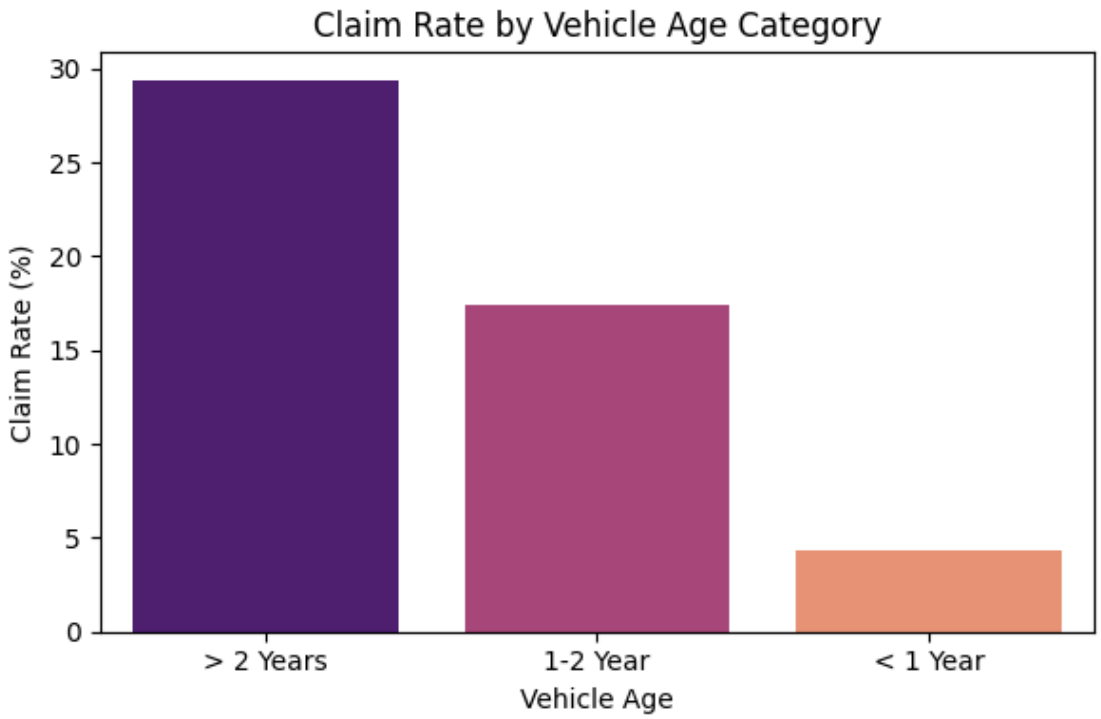
```
Vehicle_Age
1-2 Year     200316
< 1 Year     164786
> 2 Years     16007
Name: count, dtype: int64
```

```
# Calculate claim rate by vehicle age
vehicle_age_claims = data.groupby('Vehicle_Age')['Response'].agg(['count',
'sum'])
vehicle_age_claims['Claim Rate (%)'] = (vehicle_age_claims['sum'] /
vehicle_age_claims['count']) * 100
vehicle_age_claims = vehicle_age_claims.sort_values(by='Claim Rate (%)',
ascending=False)
print(vehicle_age_claims)
```

```
              count    sum  Claim Rate (%)
Vehicle_Age
> 2 Years     16007   4702       29.374649
1-2 Year     200316  34806       17.375547
< 1 Year     164786   7202        4.370517
```

```
# Barplot of 'Claim Rate by Vehicle Age Category
plt.figure(figsize=(6, 4))
sns.barplot(x=vehicle_age_claims.index, y='Claim Rate (%)',
data=vehicle_age_claims, palette='magma')
plt.title('Claim Rate by Vehicle Age Category')
plt.xlabel('Vehicle Age')
plt.ylabel('Claim Rate (%)')
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-58-4278143539.py:7: 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_claims.index, y='Claim Rate (%)',
data=vehicle_age_claims, palette='magma')
```




🔍 ***Insights :*** You May Find Older vehicles (>2 years) may show higher claim rates, possibly due to more wear and tear or maintenance issues.

Newer vehicles (<1 year) may have lower claim rates, possibly due to being in better condition or better safety features.

Mid-age vehicles (1–2 years) may fall somewhere in between.

## *Step 10. Region-wise Analysis:*

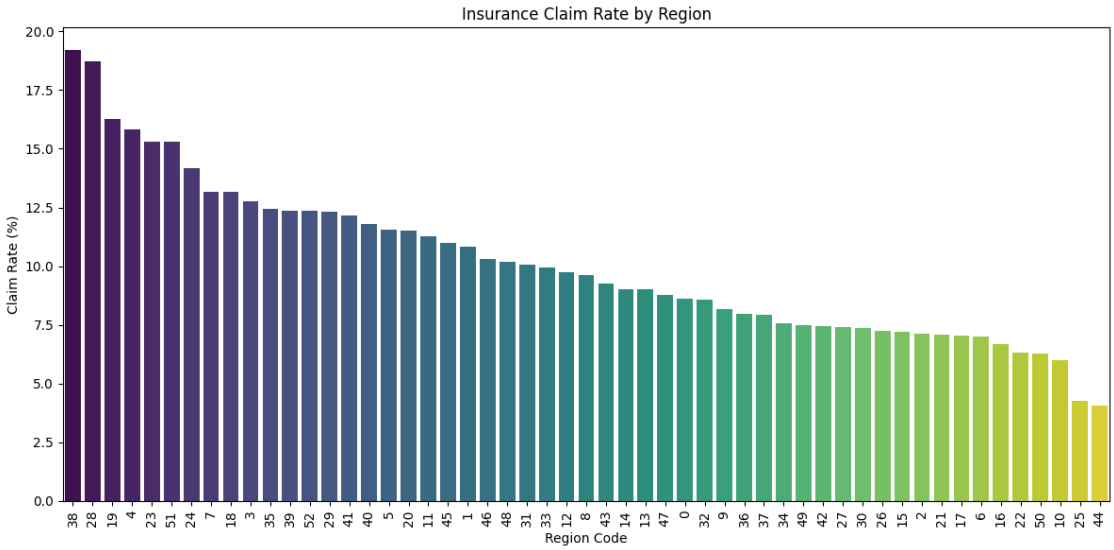
**Analyze regional patterns in insurance claims**

```
# Preview relevant columns
data[['Region_Code', 'Response']].head()
```

| | Region_Code | Response |
|---|---|---|
| 0 | 28 | 1 |
| 1 | 3 | 0 |
| 2 | 28 | 1 |
| 3 | 11 | 0 |
| 4 | 41 | 0 |

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

# Calculate claim rate by region
region_claims = data.groupby('Region_Code')['Response'].agg(['count', 'sum'])
region_claims['Claim_Rate (%)'] = (region_claims['sum'] /
region_claims['count']) * 100

# Sort by claim rate and select top regions for better visualization if
needed
region_claims = region_claims.sort_values(by='Claim_Rate (%)',
ascending=False)

plt.figure(figsize=(12, 6))
sns.barplot(x=region_claims.index.astype(str), y=region_claims['Claim_Rate
(%)'], palette='viridis')
plt.xticks(rotation=90)
plt.title('Insurance Claim Rate by Region')
plt.xlabel('Region Code')
plt.ylabel('Claim Rate (%)')
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-60-3147353967.py:12: 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_claims.index.astype(str), y=region_claims['Claim_Rate
(%)'], palette='viridis')
```




## 🔍 ***Interpretation & Insights:***

High-Claim Regions: Regions with the highest Claim Rate (%) might be more risky or have specific behaviors (e.g., urban traffic, higher vehicle density).

Volume vs. Rate: A region might have a high number of customers but a low claim rate (or vice versa).

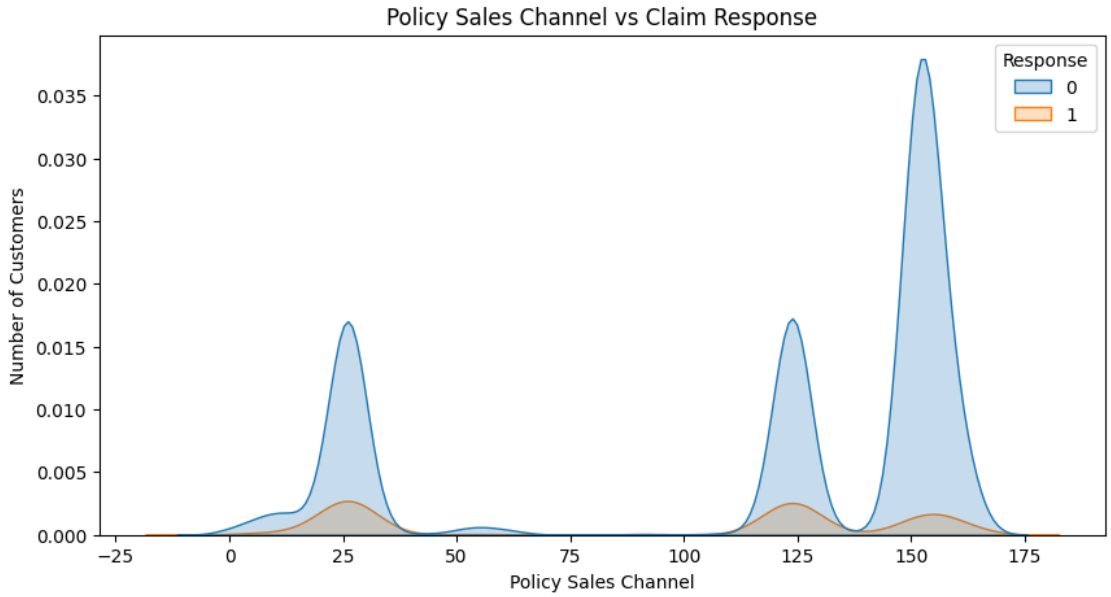
## ***Step 11. Policy Analysis:***

Explore the distribution and impact of different insurance policy types.

```
data.columns
```

```
Index(['id', 'Gender', 'Age', 'Driving_License', 'Region_Code',
       'Previously_Insured', 'Vehicle_Age', 'Vehicle_Damage',
'Annual_Premium',
       'Policy_Sales_Channel', 'Vintage', 'Response', 'Age_Group'],
      dtype='object')
```

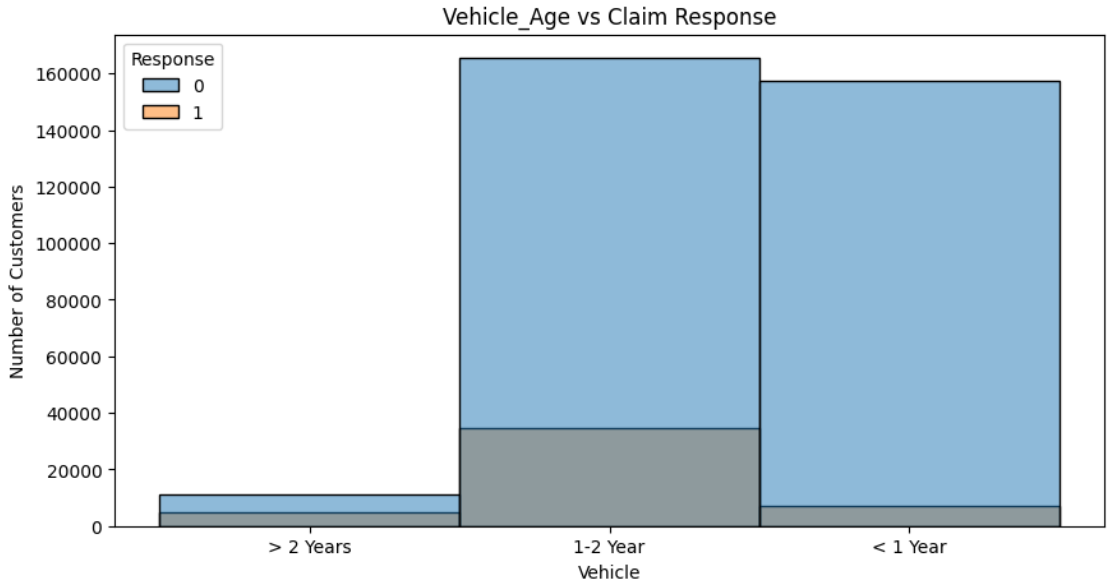
```
#Plot 1. kdeplot of Policy Sales Channel vs Claim Response
import seaborn as sns
plt.figure(figsize=(10, 5))
sns.kdeplot(data=data,x='Policy_Sales_Channel',
hue='Response',fill=True,color="salmon")
plt.title('Policy Sales Channel vs Claim Response')
plt.xlabel('Policy Sales Channel')
plt.ylabel('Number of Customers')
plt.show()
```




🧠 ***INSIGHTS:***

1. Channels with high conversion (Response = 1) may indicate effective communication or targeting.
2. Channels with lots of customers but low claims interest could suggest areas needing marketing improvement.

```
# plot 2.Histplot of Vehicle Age vs Claim Response
plt.figure(figsize=(10, 5))
sns.histplot(data=data,x='Vehicle_Age', hue='Response',color="purple")
plt.title('Vehicle_Age vs Claim Response')
plt.xlabel('Vehicle')
plt.ylabel('Number of Customers')
plt.show()
```




## ✅ *Insights from the Plot*

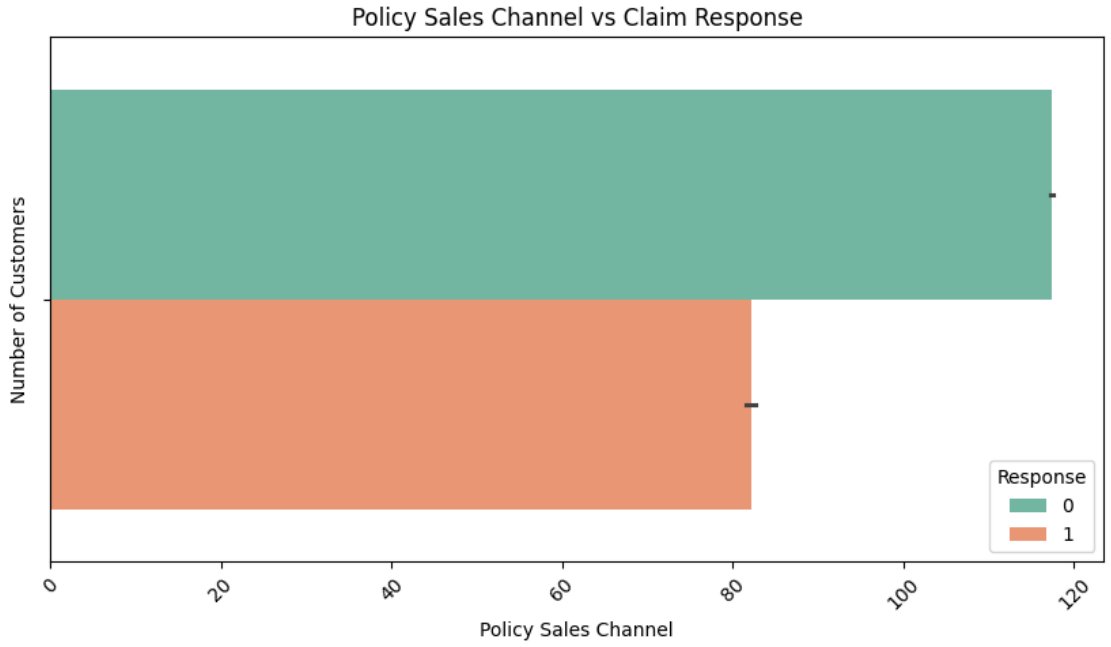
1. Customers with older vehicles (especially >2 years old) are more likely to file claims.

This could be due to: More wear and tear, More repair-prone, Lower vehicle value → more claims for minor damage

2. New vehicles are less likely to be involved in claims, possibly due to:

Better safety tech,more cautious driving,due to warranties given by the salers.

```
#Plot 3. Barplot of Policy Sales Channel vs Claim Response
top_channels = data['Policy_Sales_Channel'].value_counts().nlargest(4).index

plt.figure(figsize=(10, 5))
sns.barplot(data=data[data['Policy_Sales_Channel'].isin(top_channels)],
              x='Policy_Sales_Channel', hue='Response', palette='Set2')
plt.title('Policy Sales Channel vs Claim Response')
plt.xlabel('Policy Sales Channel')
plt.ylabel('Number of Customers')
plt.xticks(rotation=45)
plt.show()
```




✅ ***INSIGHT:***

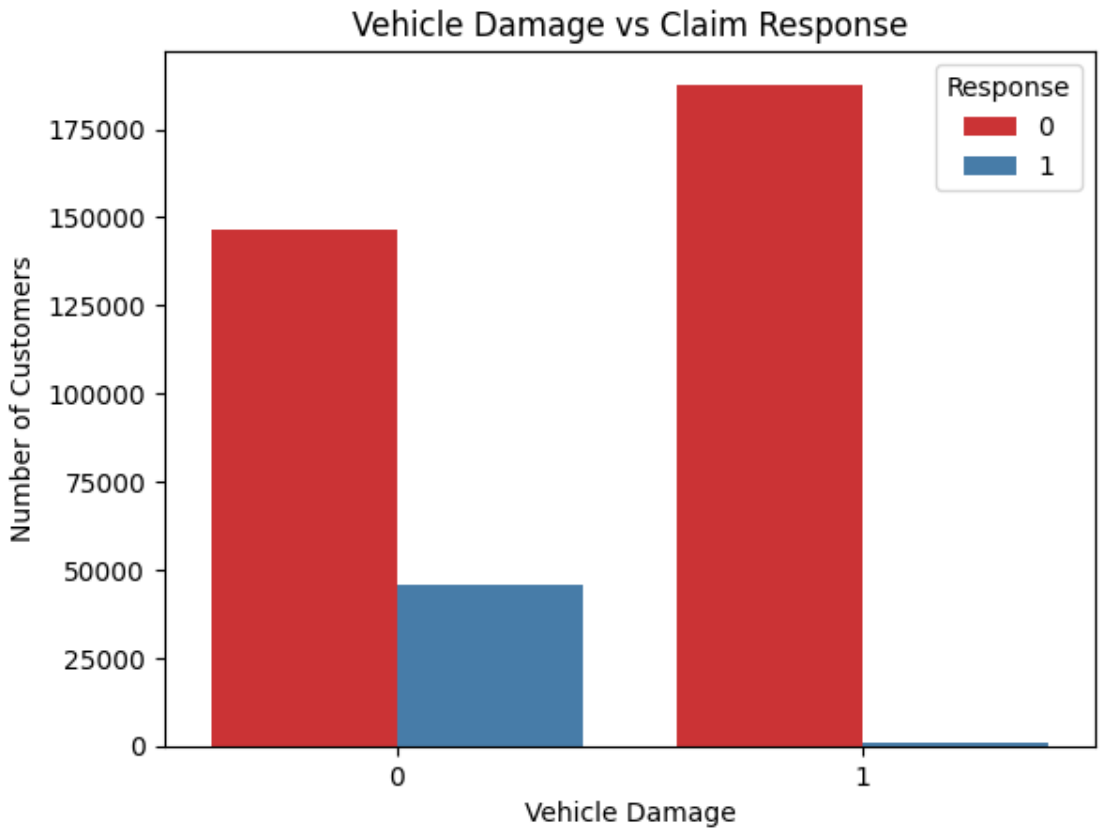
1. It shows that a particular channel that has high total customers but low claim interest, it might be overused or poorly targeted.

1. A channel with high Response = 1 ratio may indicate better lead quality, sales effectiveness, or customer awareness.

## *Step 12. Claim Frequency by Vehicle Damage:*

Investigate the relationship between vehicle damage and claim frequencies

```
# Countplot of Vehicle Damage vs Claim Response
sns.countplot(x='Vehicle_Damage', hue='Response', data=data, palette='Set1')
plt.title('Vehicle Damage vs Claim Response')
plt.xlabel('Vehicle Damage')
plt.ylabel('Number of Customers')
plt.show()
```



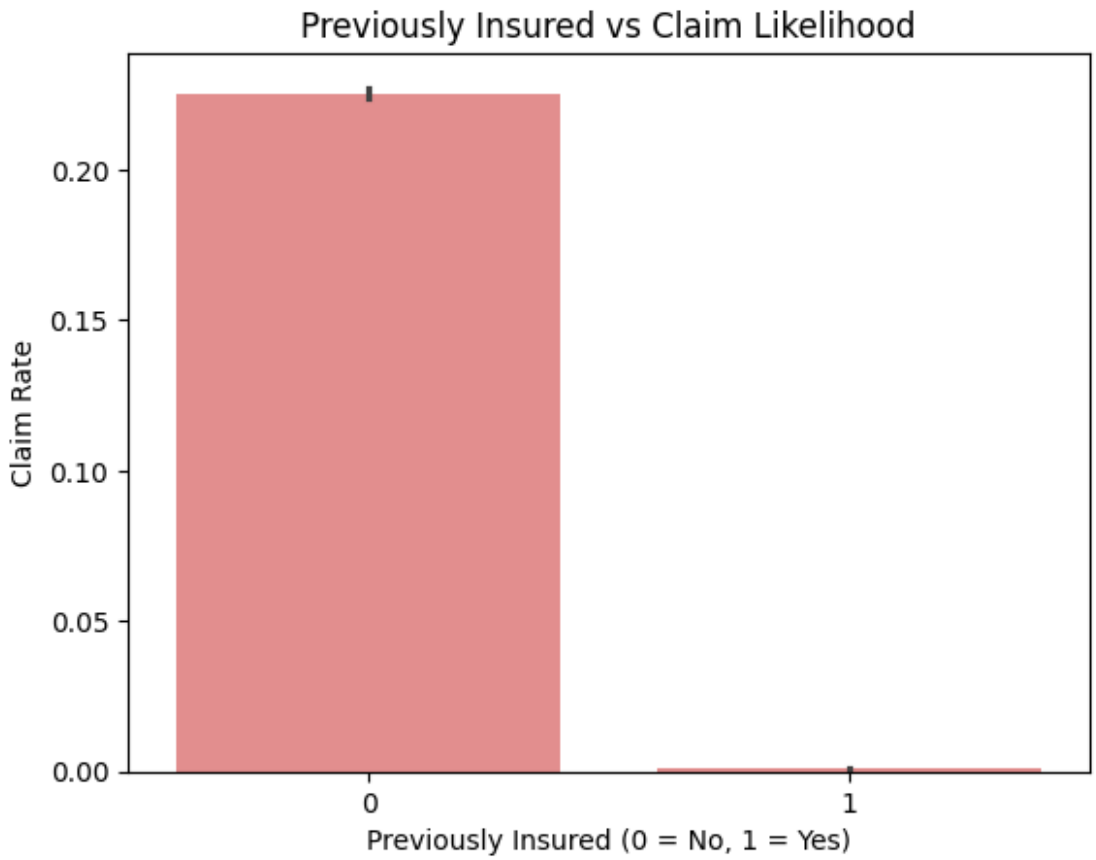

🧠 ***Interpretation of the Plot:*** For Vehicle_Damage = Yes, the count of Response = 1 (interested in policy / claim) is significantly higher than for No.

For Vehicle_Damage = No, Response = 1 is usually much lower.

## *Step 13. Customer Loyalty:*

Analyze if the number of policies held by a customer influences claim likelihood.

```
# Barplot of Previously Insured vs Claim Likelihood
sns.barplot(data=data, x='Previously_Insured',
y='Response',color="lightcoral")
plt.title('Previously Insured vs Claim Likelihood')
plt.xlabel('Previously Insured (0 = No, 1 = Yes)')
plt.ylabel('Claim Rate')
plt.show()
```




INSIGHTS: Customers with no previous insurance may have higher claim, while people who have insured previouslyare likely to claim less.

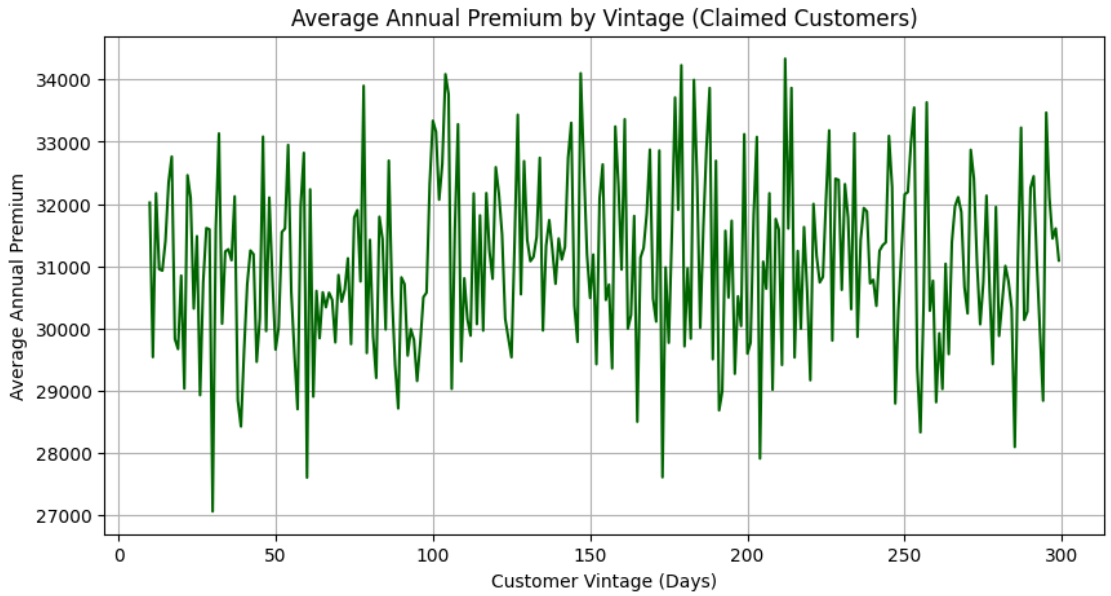
## ***Step 14. Time Analysis:***

If applicable, explore temporal patterns in insurance claims.

```
data.head()
```

{"type":"dataframe","variable_name":"data"}

```
# Group by Vintage and calculate average premium
avg_premium = data[data['Response'] ==
1].groupby('Vintage')['Annual_Premium'].mean().reset_index()

# Plot the smoothed line
plt.figure(figsize=(10, 5))
sns.lineplot(data=avg_premium, x='Vintage', y='Annual_Premium',
color='darkgreen')
plt.title('Average Annual Premium by Vintage (Claimed Customers)')
plt.xlabel('Customer Vintage (Days)')
plt.ylabel('Average Annual Premium')
plt.grid(True)
plt.show()
```




## 🧠 ***INSIGHTS:***

1. If premiums increase with tenure, it may suggest long-term customers prefer or are sold higher-value policies.

1. A drop or plateau may indicate older customers stick with standard or cheaper plans.

2. Spikes might indicate targeted upselling around specific vintage durations.

## ✅ ***Conclusion:***

## Vehicle Insurance Dataset Analysis: The exploratory analysis of the vehicle insurance dataset provided valuable insights into customer behavior, risk indicators, and claim patterns. Here are the key takeaways:

🔹 1. Imbalanced Response Variable Majority of customers did not opt for insurance (Response = 0), showing a class imbalance.

This affects modeling and requires balancing techniques for fair prediction performance.

🔹 2. Customer Age Trends Most customers fall in the early 20s to mid-40s, with varying claim tendencies.

Younger customers are more likely to opt out of insurance — possibly due to lower perceived risk or cost concerns.

🔹 3. Premium Distribution Premiums are right-skewed, with most values below ₹50,000.

High premiums may discourage customers; careful pricing strategies could improve uptake.

🔹 4. Vehicle Damage and Claims Customers who reported previous vehicle damage are more likely to claim.

This is a strong predictive feature for insurance response and risk modeling.

🔹 5. Previously Insured Customers Those who were previously insured are less likely to claim again, possibly due to safer driving or better awareness.

Indicates a trustworthy customer segment for targeted retention.

🔹 6. Regional Patterns Certain Region_Codes have higher claim rates — regional segmentation can help in personalized offers or risk-adjusted pricing.