In [3]:
```
import pandas as pd
```

In [4]:
```
data = pd.read_csv("C:/Users/NITISH/Downloads/INSURANCE.csv", encoding='lat
print(data)
```

```
     Gender       City Age_group          Occupation     Monthly_Income  \
0      Male     Mumbai     31-40        Own Business       INR 76K-100K
1    Female  Bangalore     41-50  Salaried  Private       INR 76K-100K
2      Male  Ahmedabad     41-50        Own Business        INR 51K-75K
3      Male  Bangalore     26-30  Salaried  Private        INR 51K-75K
4      Male  Ahmedabad     51-60        Own Business        INR 51K-75K
..      ...        ...       ...                 ...                ...
335    Male    Lucknow     18-25             Student  Less than INR 20K
336    Male  Bangalore     18-25  Salaried  Private        INR 21K-35K
337    Male  Bangalore     18-25  Salaried  Private        INR 36K-50K
338    Male      Delhi     18-25  Salaried  Private  Less than INR 20K
339    Male  Ahmedabad     18-25  Salaried  Private  Less than INR 20K

     Life_Insurance             Health_Insurance  Auto_Insurance  \
0    Life Insurance                          NAN             NAN
1    Life Insurance  Health Insurance (Mediclaim)  Auto Insurance
2    Life Insurance  Health Insurance (Mediclaim)  Auto Insurance
3    Life Insurance  Health Insurance (Mediclaim)             NAN
4               NAN                          NAN  Auto Insurance
..              ...                          ...             ...
335             NAN                          NAN  Auto Insurance
336  Life Insurance  Health Insurance (Mediclaim)  Auto Insurance
337  Life Insurance                          NAN  Auto Insurance
338             NAN  Health Insurance (Mediclaim)  Auto Insurance
339             NAN                          NAN  Auto Insurance

              Life_Insurance_Buy          Health_Insurance_Buy  ...  \
0       Online- from Policybazaar                             -  ...
1           Offline- via an agent        Offline- via an agent  ...
2           Offline- via an agent        Offline- via an agent  ...
3           Offline- via an agent        Offline- via an agent  ...
4                               -                             -  ...
..                            ...                           ...  ...
335                             -                             -  ...
336  Online- from company website  Online- from company website  ...
337  Online- from company website                             -  ...
338                             -        Offline- via an agent  ...
339                             -                             -  ...

    ICICI_Lombard_AI Niva_Bupa_(Max_Bupa)_AI LIC_AI Aditya_Birla_AI  \
0                  -                       -      -               -
1      ICICI Lombard                       -      -               -
2                  -                       -      -               -
3                  -                       -      -               -
4                  -                       -      -               -
..               ...                     ...    ...             ...
335                -                       -      -               -
336                -                       -      -               -
337                -                       -      -               -
338                -                       -      -               -
339                -                       -      -               -

    Bajaj_Allianz_AI Acko_AI Kotak_Mahindra_AI Digit_AI Others_AI Others_A
I.1
0                  -       -                 -        -         -
NaN
1                  -       -                 -        -         -
NaN
2                  -       -                 -        -         -
NaN
3                  -       -                 -        -         -
```

```
NaN
4                   -    Acko                 -        -         -
NaN
..                ...     ...               ...      ...       ...
...
335                 -       -                 -    Digit         -
NaN
336                 -       -                 -        -       OLX
NaN
337    Bajaj Allianz       -                 -        -         -
NaN
338                 -       -                 -        -         -
NaN
339    Bajaj Allianz       -                 -        -         -
NaN

[340 rows x 44 columns]
```

In [5]:
```
gender_distribution = data['Gender'].value_counts()
city_distribution = data['City'].value_counts()
age_group_distribution = data['Age_group'].value_counts()
occupation_distribution = data['Occupation'].value_counts()
```

In [6]:
```
data['Monthly_Income'] = pd.Categorical(data['Monthly_Income'], ordered=Tru
    'Less than INR 25K', 'INR 25K-50K', 'INR 51K-75K', 'INR 76K-100K', 'Mor
])
monthly_income_distribution = data['Monthly_Income'].value_counts()
```

In [7]:
```
life_insurance_ownership = data['Life_Insurance'].value_counts(normalize=Tr
health_insurance_ownership = data['Health_Insurance'].value_counts(normaliz
auto_insurance_ownership = data['Auto_Insurance'].value_counts(normalize=Tr
```

In [8]:
```
life_insurance_purchase_channel = data['Life_Insurance_Buy'].value_counts(n
health_insurance_purchase_channel = data['Health_Insurance_Buy'].value_coun
auto_insurance_purchase_channel = data['Auto_Insurance_Buy'].value_counts(n
```

In [9]:
```
life_insurance_companies = ['HDFC_Life_LI', 'ICICI_Prudential_LI', 'Niva_Bu
                        'SBI_Life_LI', 'Bajaj_Allianz_LI', 'Bharti_Axa_LI',
                        'Tata_AIA_LI', 'Other_LI']
life_insurance_company_preferences = data[life_insurance_companies].apply(p
```

In [10]:
```
correlation_age_income = data[['Age_group', 'Monthly_Income']].groupby('Age
```

In [11]:
```
summary_statistics = data.describe()
```

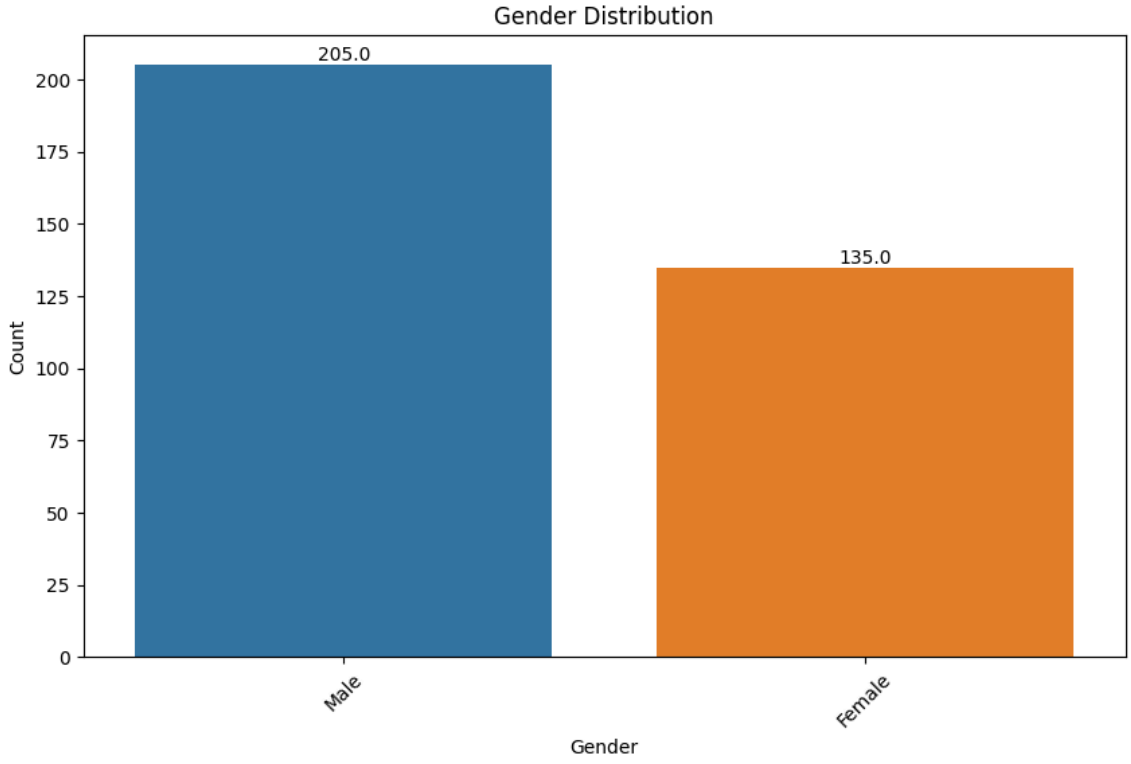
In [12]:
```
print("Gender Distribution:\n", gender_distribution)
print("\nCity Distribution:\n", city_distribution)
print("\nAge Group Distribution:\n", age_group_distribution)
print("\nOccupation Distribution:\n", occupation_distribution)
print("\nMonthly Income Distribution:\n", monthly_income_distribution)
print("\nLife Insurance Ownership:\n", life_insurance_ownership)
print("\nHealth Insurance Ownership:\n", health_insurance_ownership)
print("\nAuto Insurance Ownership:\n", auto_insurance_ownership)
print("\nLife Insurance Purchase Channel:\n", life_insurance_purchase_chann
print("\nHealth Insurance Purchase Channel:\n", health_insurance_purchase_c
print("\nAuto Insurance Purchase Channel:\n", auto_insurance_purchase_chann
print("\nCorrelation between Age and Monthly Income:\n", correlation_age_in
print("\nSummary Statistics:\n", summary_statistics)
```

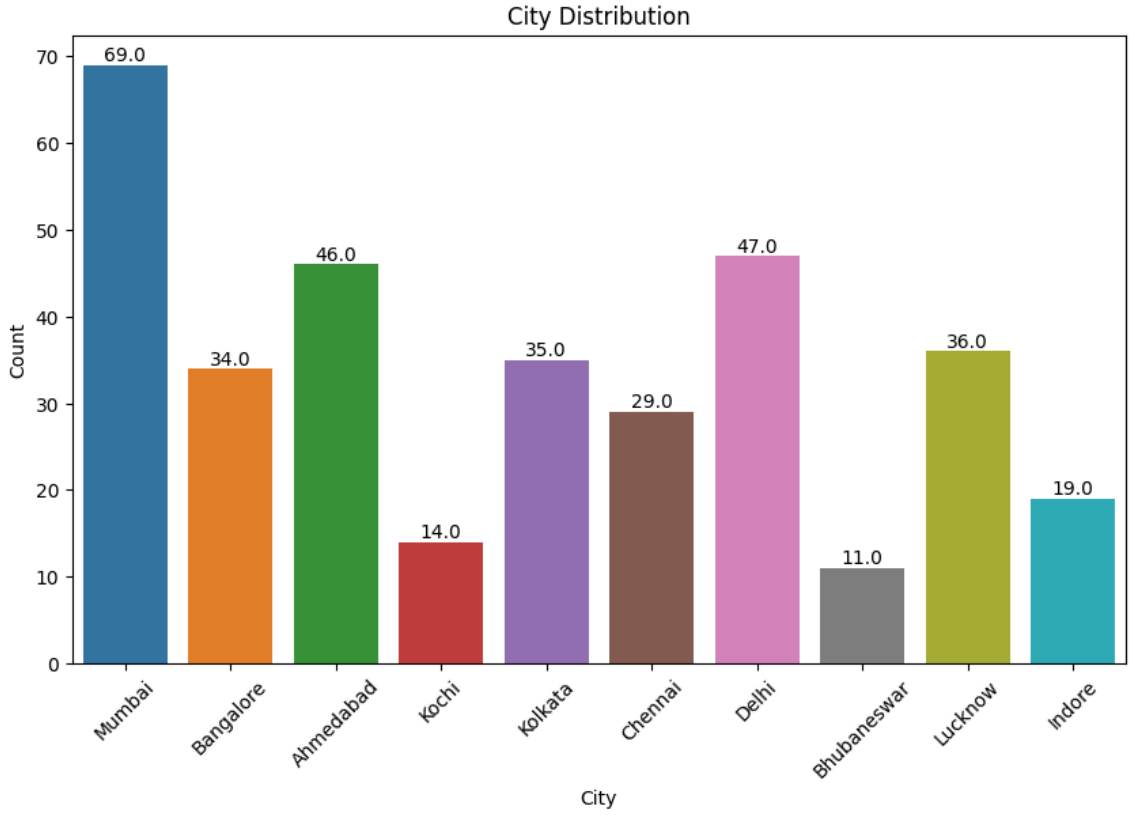
```
Gender Distribution:
 Gender
Male      205
Female    135
Name: count, dtype: int64

City Distribution:
 City
Mumbai         69
Delhi          47
Ahmedabad      46
Lucknow        36
Kolkata        35
Bangalore      34
Chennai        29
Indore         19
Kochi          14
Bhubaneswar    11
Name: count, dtype: int64
```

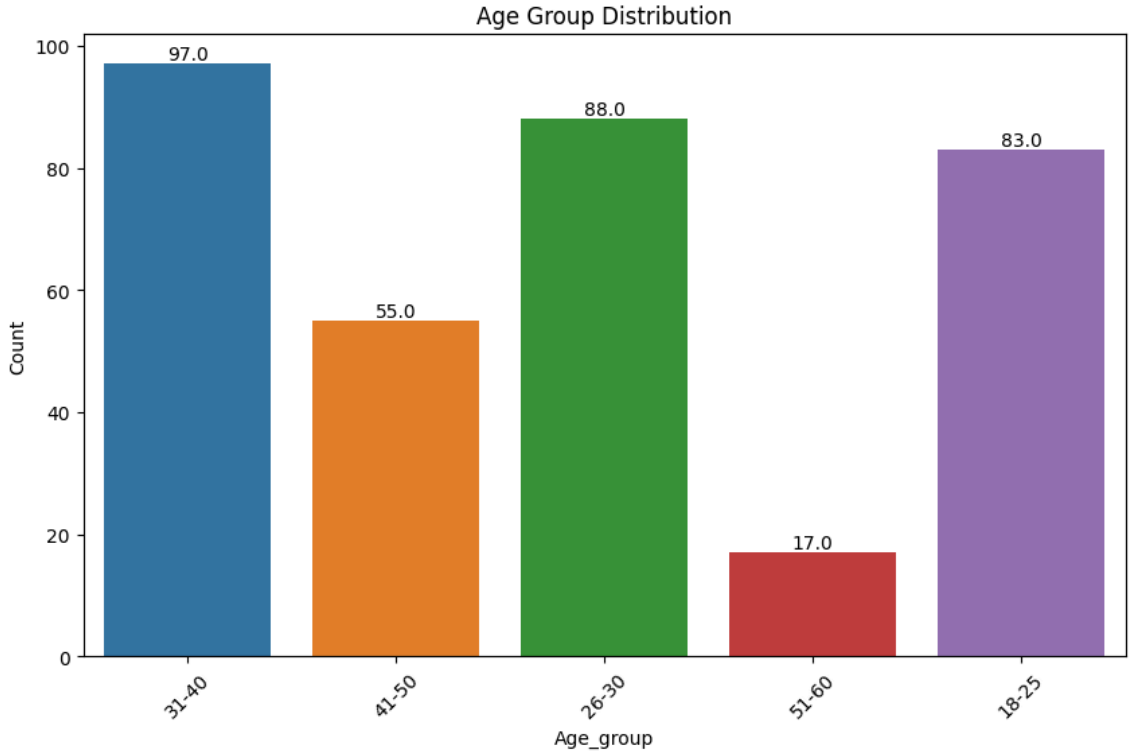
In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [14]:
```
def plot_count_with_annotation(data, x, title):
    plt.figure(figsize=(10, 6))
    plot = sns.countplot(data=data, x=x)
    plt.title(title)
    plt.xlabel(x)
    plt.ylabel('Count')
    plt.xticks(rotation=45)
    for p in plot.patches:
        plot.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2.,
                      ha='center', va='center', xytext=(0, 5), textcoords='
    plt.show()
```

In [15]:
```
plot_count_with_annotation(data, 'Gender', 'Gender Distribution')
plot_count_with_annotation(data, 'City', 'City Distribution')
plot_count_with_annotation(data, 'Age_group', 'Age Group Distribution')
```







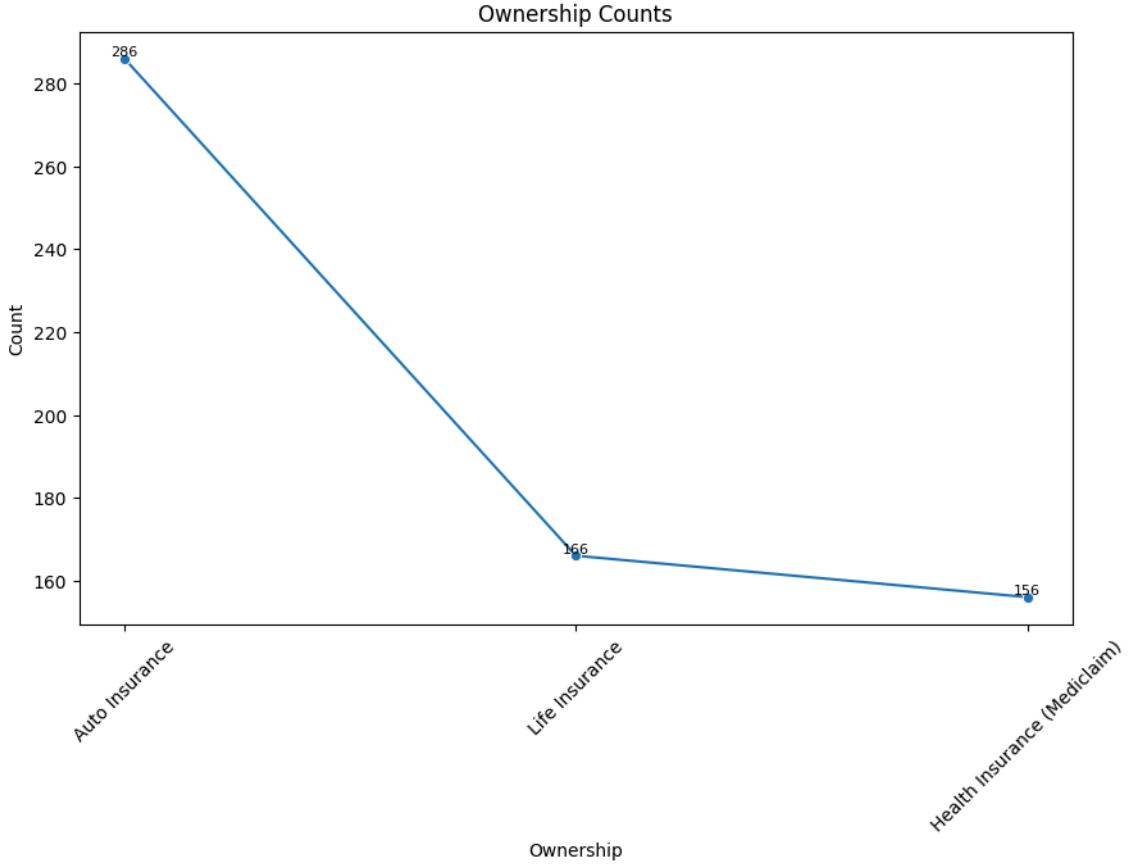
In [16]:
```
unique_values = data['Life_Insurance'].unique()
print(unique_values)
unique_values = data['Health_Insurance'].unique()
print(unique_values)
unique_values = data['Auto_Insurance'].unique()
print(unique_values)
```

```
['Life Insurance' 'NAN']
['NAN' 'Health Insurance (Mediclaim)']
['NAN' 'Auto Insurance']
```

In [17]:
```
import numpy as np
data.replace('NAN', np.nan, inplace=True)
```

In [18]:
```
# Function to plot count and annotate on each point
def plot_count_with_annotation(data, x, title):
    plt.figure(figsize=(10, 6))
    plot = sns.lineplot(data=data, x=x, y='Count', marker='o')
    plt.title(title)
    plt.xlabel(x)
    plt.ylabel('Count')
    plt.xticks(rotation=45)
    for index, row in data.iterrows():
        plot.text(row[x], row['Count'], str(row['Count']), ha='center', va=
    plt.show()

# Insurance Ownership
insurance_data = pd.melt(data[['Life_Insurance', 'Health_Insurance', 'Auto_

# Count the occurrences of each ownership type
count_data = insurance_data['Ownership'].value_counts().reset_index()
count_data.columns = ['Ownership', 'Count']

# Remove rows where Ownership value is NaN
count_data = count_data[count_data['Ownership'].notnull()]

plot_count_with_annotation(count_data, 'Ownership', 'Ownership Counts')
```

```
C:\Users\NITISH\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a fut
ure version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\NITISH\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a fut
ure version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

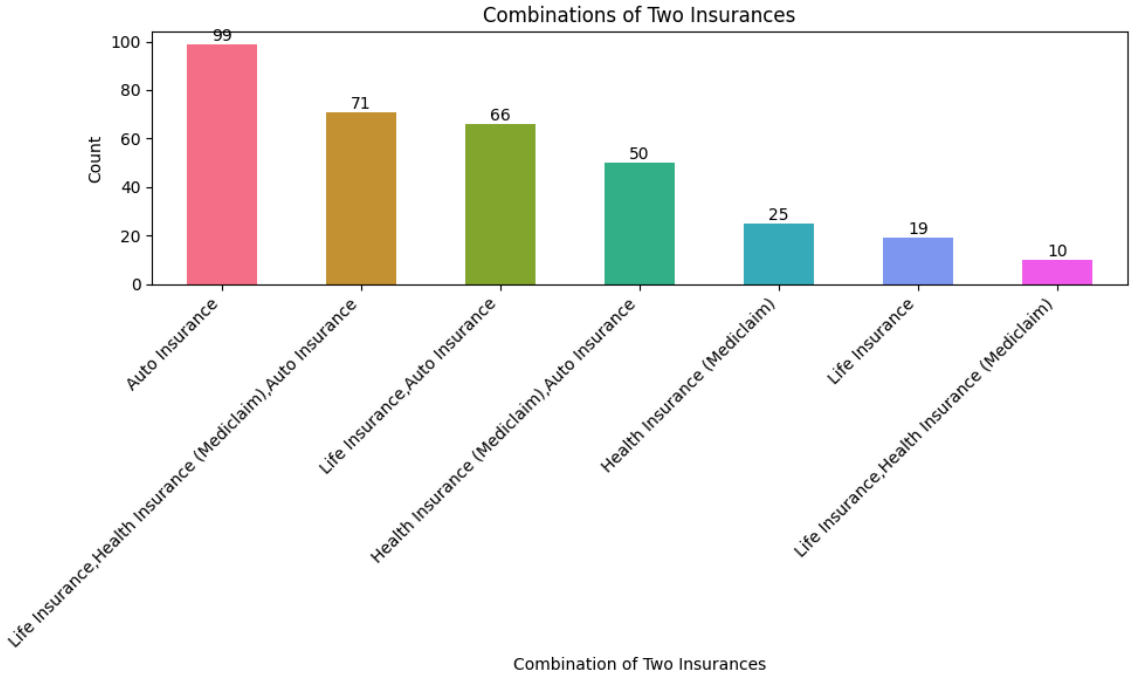


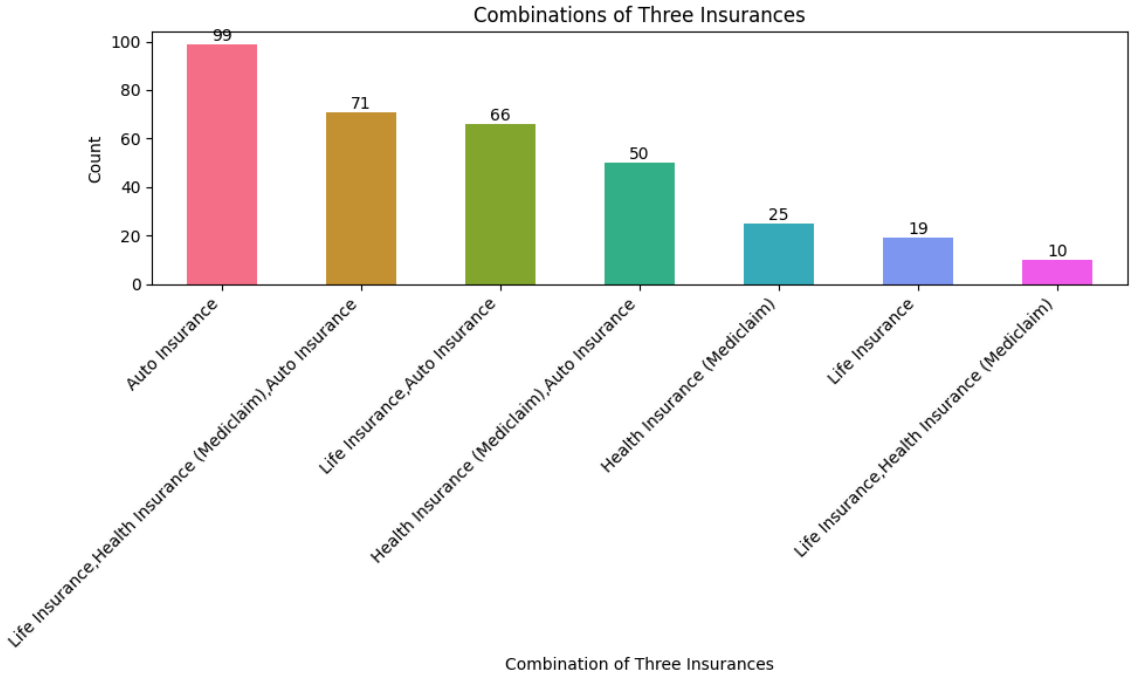
In [21]:
```
data.replace('NAN', np.nan, inplace=True)

# Count the occurrences of each combination of insurance ownership
data['Num_Insurances'] = data[['Life_Insurance', 'Health_Insurance', 'Auto_

# Count the number of people with 2 insurances
num_people_2_ins = (data['Num_Insurances'] == 2).sum()

# Count the number of people with all 3 insurances
num_people_all_ins = (data['Num_Insurances'] == 3).sum()

print(f"Number of people with 2 insurances: {num_people_2_ins}")
print(f"Number of people with all 3 insurances: {num_people_all_ins}")
```

```
Number of people with 2 insurances: 126
Number of people with all 3 insurances: 71
```

In [22]:
```
# print(two_insurance_counts)
# print(three_insurance_counts)
```

In [23]:
```
import seaborn as sns

data.replace('NAN', np.nan, inplace=True)

# Create new columns indicating the combinations of insurances
data['Two_Insurances'] = data[['Life_Insurance', 'Health_Insurance', 'Auto_
data['Three_Insurances'] = data[['Life_Insurance', 'Health_Insurance', 'Aut

# Count the occurrences of each combination of two insurances
two_insurance_counts = data['Two_Insurances'].value_counts()
# Count the occurrences of each combination of three insurances
three_insurance_counts = data['Three_Insurances'].value_counts()

# Define custom color palettes
colors_two = sns.color_palette("husl", len(two_insurance_counts))
colors_three = sns.color_palette("husl", len(three_insurance_counts))

# Plotting bar charts for combinations of two and three insurances with cus
plt.figure(figsize=(10, 6))
two_insurance_counts.plot(kind='bar', color=colors_two)
plt.title('Combinations of Two Insurances')
plt.xlabel('Combination of Two Insurances')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

# Add count on each bar
for i, count in enumerate(two_insurance_counts):
    plt.text(i, count, str(count), ha='center', va='bottom')

plt.show()

plt.figure(figsize=(10, 6))
three_insurance_counts.plot(kind='bar', color=colors_three)
plt.title('Combinations of Three Insurances')
plt.xlabel('Combination of Three Insurances')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

# Add count on each bar
for i, count in enumerate(three_insurance_counts):
    plt.text(i, count, str(count), ha='center', va='bottom')

plt.show()
```


Combinations of Two Insurances
Count
99
71
66
50
25
19
10
Auto Insurance
Life Insurance,Health Insurance (Mediclaim),Auto Insurance
Life Insurance,Auto Insurance
Health Insurance (Mediclaim),Auto Insurance
Health Insurance (Mediclaim)
Life Insurance
Life Insurance,Health Insurance (Mediclaim)
Combination of Two Insurances


Combinations of Three Insurances
Count
99
71
66
50
25
19
10
Auto Insurance
Life Insurance,Health Insurance (Mediclaim),Auto Insurance
Life Insurance,Auto Insurance
Health Insurance (Mediclaim),Auto Insurance
Health Insurance (Mediclaim)
Life Insurance
Life Insurance,Health Insurance (Mediclaim)
Combination of Three Insurances

In [26]:
```
print(data.columns)
```

```
Index(['Gender', 'City', 'Age_group', 'Occupation', 'Monthly_Income',
       'Life_Insurance', 'Health_Insurance', 'Auto_Insurance',
       'Life_Insurance_Buy', 'Health_Insurance_Buy', 'Auto_Insurance_Buy',
       'HDFC_Life_LI', 'ICICI_Prudential_LI', 'Niva_Bupa_(Max_Bupa)_LI',
       'LIC_LI', 'SBI_Life_LI', 'Bajaj_Allianz_LI', 'Bharti_Axa_LI',
       'Kotak_Mahindra_LI', 'Tata_AIA_LI', 'Other_LI', 'HDFC_Ergo_HI',
       'Star_HI', 'ICICI_Lombard_HI', 'Niva_Bupa_(Max_Bupa)_HI', 'LIC_HI',
       'Aditya_Birla_HI', 'Bajaj_Allianz_HI', 'Acko_HI', 'Kotak_Mahindra_H
I',
       'Digit_HI', 'Others_HI', 'HDFC_Ergo_AI', 'Star_AI', 'ICICI_Lombard_
AI',
       'Niva_Bupa_(Max_Bupa)_AI', 'LIC_AI', 'Aditya_Birla_AI',
       'Bajaj_Allianz_AI', 'Acko_AI', 'Kotak_Mahindra_AI', 'Digit_AI',
       'Others_AI', 'Others_AI.1', 'Num_Insurances', 'Two_Insurances',
       'Three_Insurances'],
      dtype='object')
```

In [27]:
```
pip install xgboost
```

```
Requirement already satisfied: xgboost in c:\users\nitish\anaconda3\lib\si
te-packages (2.0.3)Note: you may need to restart the kernel to use updated
packages.

Requirement already satisfied: numpy in c:\users\nitish\anaconda3\lib\site
-packages (from xgboost) (1.24.3)
Requirement already satisfied: scipy in c:\users\nitish\anaconda3\lib\site
-packages (from xgboost) (1.11.1)
```

In [28]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score
```

In [29]:
```
data1={
    'Age_Group': ['26-30', '41-50', '18-25', '26-30', '41-50', '18-25', '26
    'Gender': ['Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male'
    'City': ['Mumbai', 'Delhi', 'Bangalore', 'Kolkata', 'Chennai', 'Hyderab
    'HDFC_Life_LI': [0, 0, 1, 0, 0, 0, 0, 0, 0, 0],  # Example data for ins
    'ICICI_Prudential_LI': [0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
    'Niva_Bupa_(Max_Bupa)_LI': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    'LIC_LI': [1, 0, 0, 1, 0, 1, 1, 1, 0, 0],
    'SBI_Life_LI': [0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
    'Bajaj_Allianz_LI': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    'Bharti_Axa_LI': [1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    'Kotak_Mahindra_LI': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    'Tata_AIA_LI': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    'Other_LI': [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
}
```

In [30]:
```
df1 = pd.DataFrame(data1)
```

In [31]:
```
label_encoders = {}
for col in ['Gender', 'City']:
    label_encoders[col] = LabelEncoder()
    df1[col] = label_encoders[col].fit_transform(df1[col])
```

In [32]:
```
age_group_mapping = {
    '18-25': 1,
    '26-30': 2,
    '31-40': 3,
    '41-50':4,
    '51-60':5
}
df1['Age_Group'] = df1['Age_Group'].map(age_group_mapping)
```

In [33]:
```
X = df1[['Age_Group', 'Gender', 'City']]
y = df1.drop(['Age_Group', 'Gender', 'City'], axis=1)
```

In [34]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ra
```

In [35]:
```
models = {}
for column in y.columns:
    model = XGBClassifier()
    model.fit(X_train, y_train[column])
    models[column] = model
```

In [36]:
```
y_preds = {}
for column, model in models.items():
    y_preds[column] = model.predict(X_test)
```

In [37]:
```
accuracies = {}
for column in y.columns:
    accuracies[column] = accuracy_score(y_test[column], y_preds[column])
    print(f"Accuracy for {column}: {accuracies[column]}")
```

```
Accuracy for HDFC_Life_LI: 1.0
Accuracy for ICICI_Prudential_LI: 0.5
Accuracy for Niva_Bupa_(Max_Bupa)_LI: 1.0
Accuracy for LIC_LI: 0.0
Accuracy for SBI_Life_LI: 1.0
Accuracy for Bajaj_Allianz_LI: 1.0
Accuracy for Bharti_Axa_LI: 1.0
Accuracy for Kotak_Mahindra_LI: 1.0
Accuracy for Tata_AIA_LI: 1.0
Accuracy for Other_LI: 1.0
```

1.0 insurance achieved perfect accuracy (100%) on the test data, indicating that it

correctly predicted all instances where a person prefers Life insurance. ¶

In [38]:
```
example_data = {'Age_Group': [2], 'Gender': ['Male'], 'City': ['Mumbai']}
example_df = pd.DataFrame(example_data)
for col in ['Gender', 'City']:
    example_df[col] = label_encoders[col].transform(example_df[col])

predictions = {}
for column, model in models.items():
    prediction = model.predict(example_df)
    predictions[column] = prediction

predicted_insurance_companies = [col for col, prediction in predictions.ite
print("Predicted Preferred Insurance Companies:", predicted_insurance_compa
```

```
Predicted Preferred Insurance Companies: ['LIC_LI']
```

In [ ]:
```

```