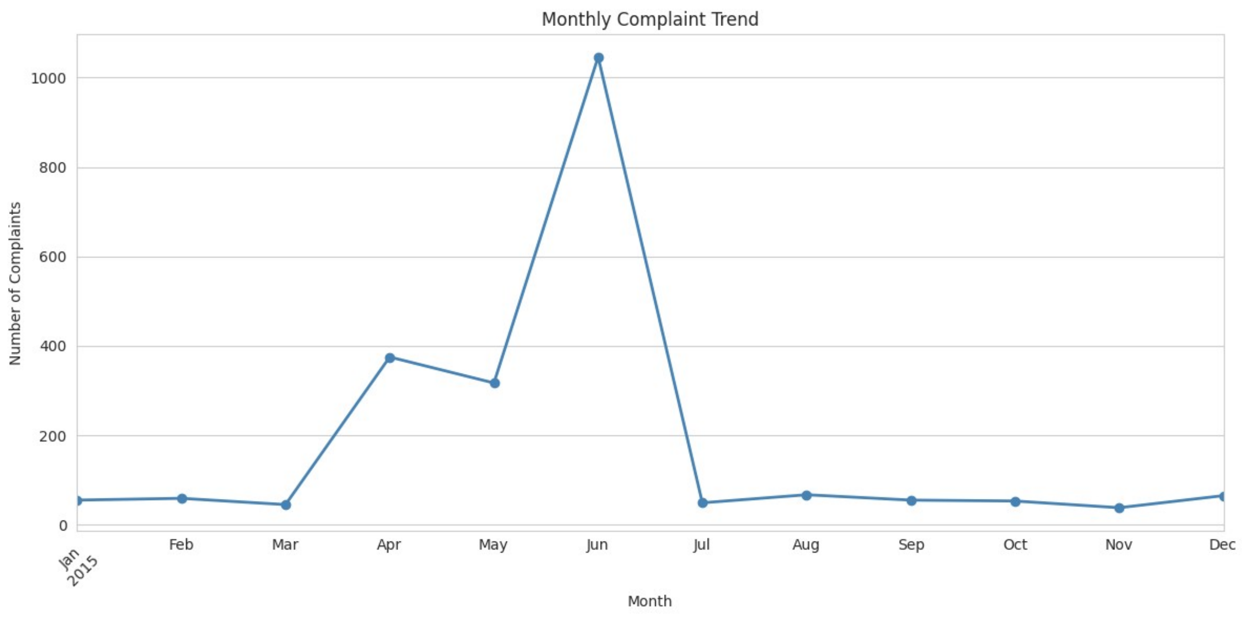
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("whitegrid")

# Load data
df = pd.read_csv('Comcast.csv')

print('Total Complaints:', len(df))
print('Dataset Shape:', df.shape)
```

```
Total Complaints: 2224
Dataset Shape: (2224, 11)
```

```python
# TASK 1: Monthly Trend
print('\nTASK 1: Monthly Trend')

df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%y',
errors='coerce')
df['Month_Year'] = df['Date'].dt.to_period('M')
monthly_complaints = df['Month_Year'].value_counts().sort_index()

print(monthly_complaints)

plt.figure(figsize=(12, 6))
monthly_complaints.plot(kind='line', marker='o', color='steelblue',
linewidth=2)
plt.title('Monthly Complaint Trend')
plt.xlabel('Month')
plt.ylabel('Number of Complaints')
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig('1_monthly_trend.png')
plt.show()

print('Peak month:', monthly_complaints.idxmax(), 'with',
monthly_complaints.max(), 'complaints')
```

```
TASK 1: Monthly Trend
Month_Year
2015-01      55
2015-02      59
2015-03      45
2015-04     375
2015-05     317
2015-06    1046
2015-07      49
```

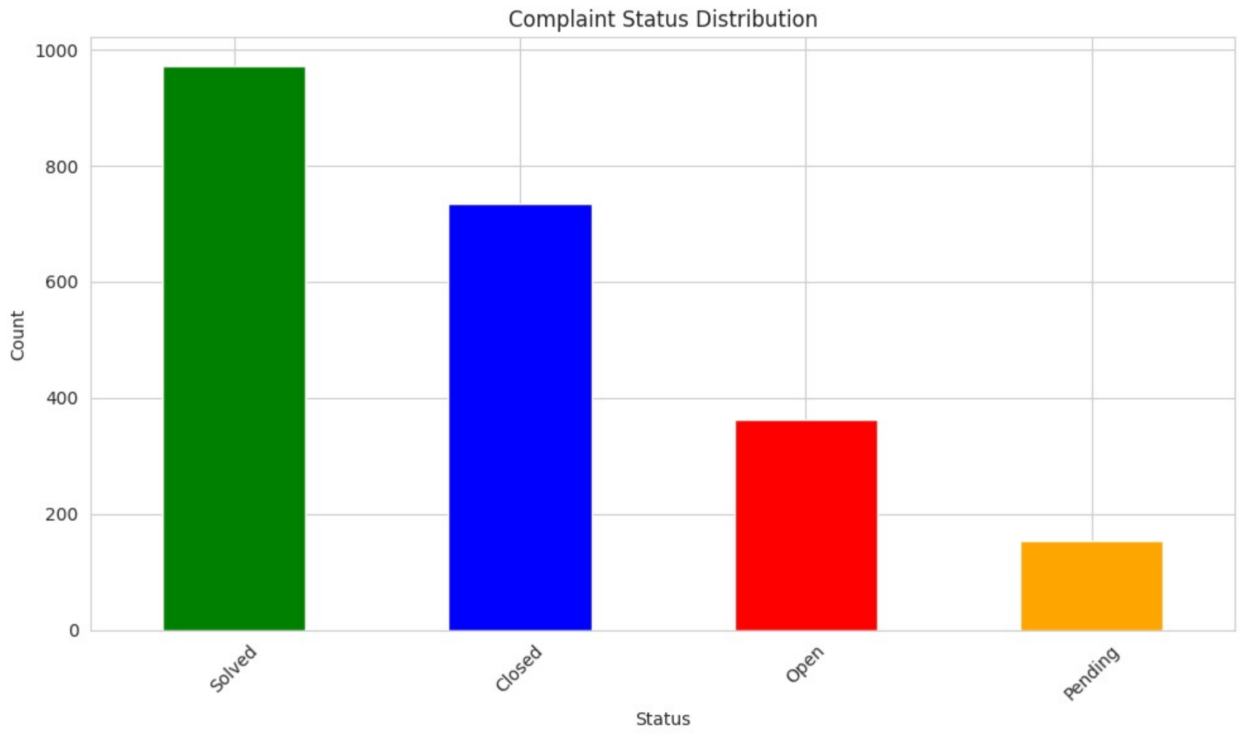
```
2015-08      67
2015-09      55
2015-10      53
2015-11      38
2015-12      65
Freq: M, Name: count, dtype: int64
```



```
Peak month: 2015-06 with 1046 complaints

# TASK 2: Status Frequency
print('\nTASK 2: Status Frequency')

status_freq = df['Status'].value_counts()
status_pct = (df['Status'].value_counts(normalize=True) *
100).round(2)

status_table = pd.DataFrame({
    'Status': status_freq.index,
    'Frequency': status_freq.values,
    'Percentage': status_pct.values
})

print(status_table.to_string(index=False))

plt.figure(figsize=(10, 6))
status_freq.plot(kind='bar', color=['green', 'blue', 'red', 'orange'])
plt.title('Complaint Status Distribution')
plt.xlabel('Status')
plt.ylabel('Count')
plt.xticks(rotation=45)
```

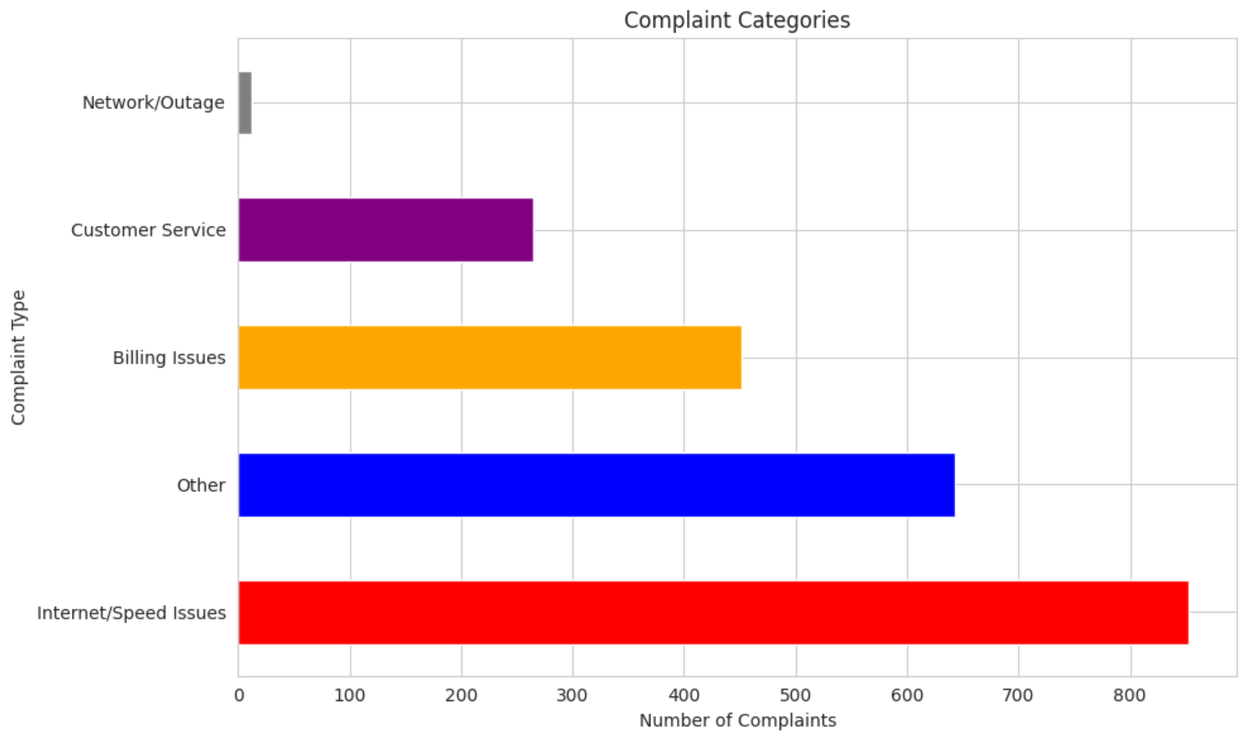
```python
plt.tight_layout()
plt.savefig('2_status_frequency.png')
plt.show()
```

```
TASK 2: Status Frequency
 Status  Frequency  Percentage
 Solved        973       43.75
 Closed        734       33.00
   Open        363       16.32
Pending        154        6.92
```

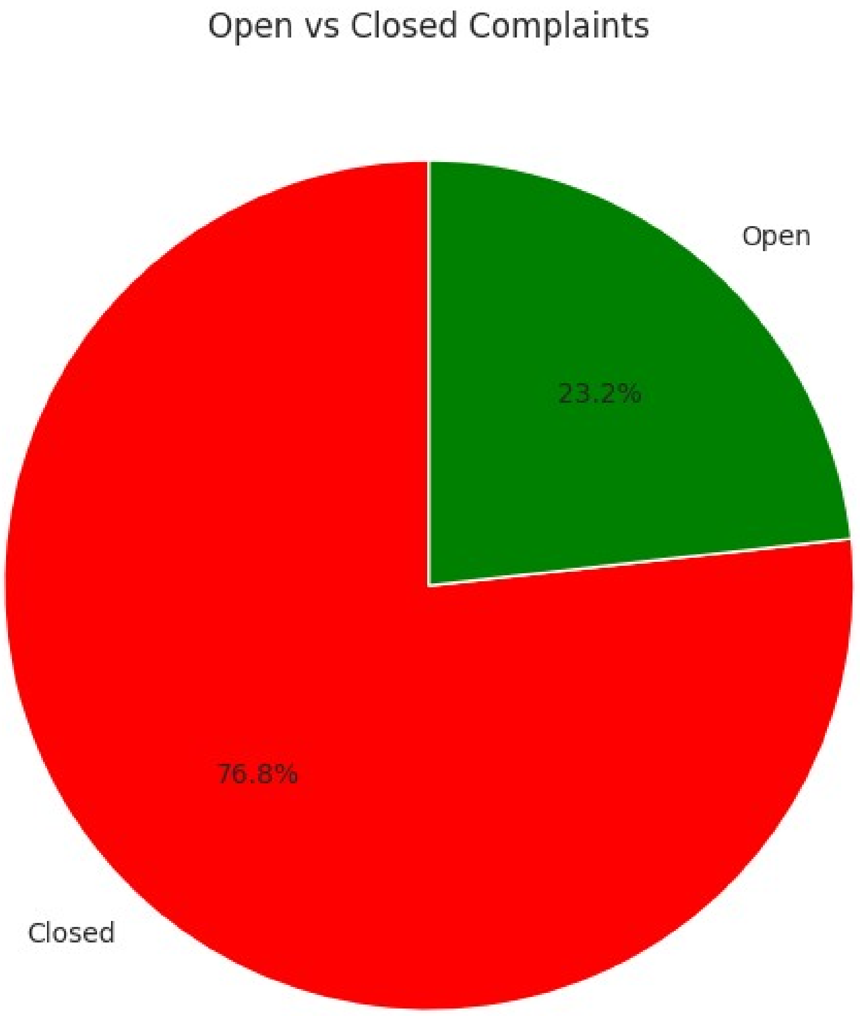


```python
# TASK 3: Complaint Types
print('\nTASK 3: Complaint Types')

def categorize_complaint(complaint):
    complaint_lower = str(complaint).lower()

    if any(word in complaint_lower for word in ['internet', 'speed',
'connection', 'bandwidth', 'data', 'wifi']):
        return 'Internet/Speed Issues'
    elif any(word in complaint_lower for word in ['billing', 'bill',
'charge', 'payment', 'price']):
        return 'Billing Issues'
    elif any(word in complaint_lower for word in ['service',
'customer', 'support']):
        return 'Customer Service'
    elif any(word in complaint_lower for word in ['network', 'outage',
'down']):
        return 'Network/Outage'
    else:
        return 'Other'

df['Complaint_Type'] = df['Customer
Complaint'].apply(categorize_complaint)
complaint_types = df['Complaint_Type'].value_counts()

print(complaint_types)
print('\nPercentage:')
print((complaint_types / len(df) * 100).round(2))

plt.figure(figsize=(10, 6))
complaint_types.plot(kind='barh', color=['red', 'blue', 'orange',
'purple', 'gray'])
plt.title('Complaint Categories')
plt.xlabel('Number of Complaints')
plt.ylabel('Complaint Type')
plt.tight_layout()
plt.savefig('3_complaint_types.png')
plt.show()

print('Maximum complaint type:', complaint_types.index[0], 'with',
complaint_types.values[0], 'complaints')
```

```
TASK 3: Complaint Types
Complaint_Type
Internet/Speed Issues    853
Other                    643
Billing Issues           451
Customer Service         265
Network/Outage            12
Name: count, dtype: int64

Percentage:
Complaint_Type
Internet/Speed Issues    38.35
Other                    28.91
Billing Issues           20.28
Customer Service         11.92
Network/Outage            0.54
Name: count, dtype: float64
```



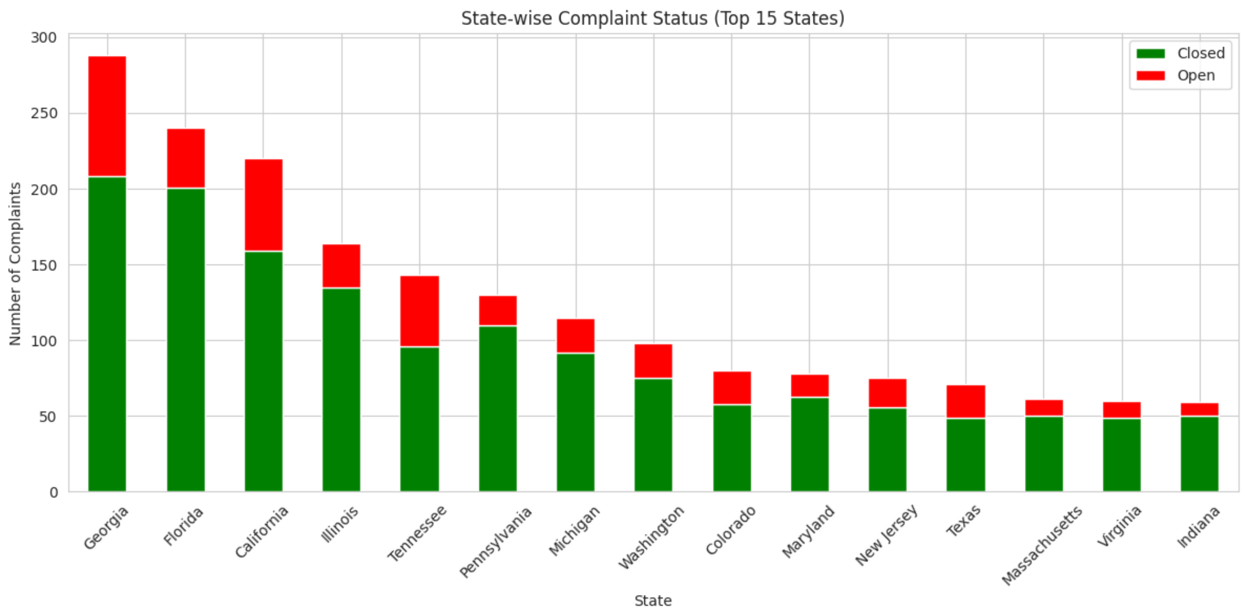
```
Maximum complaint type: Internet/Speed Issues with 853 complaints

# TASK 4: Open vs Closed
print('\nTASK 4: Open vs Closed Status')

def categorize_status(status):
    if status in ['Open', 'Pending']:
        return 'Open'
    elif status in ['Closed', 'Solved']:
        return 'Closed'
    else:
        return 'Other'

df['Status_Category'] = df['Status'].apply(categorize_status)
category_counts = df['Status_Category'].value_counts()

print(category_counts)
print('\nPercentage:')
print((category_counts / len(df) * 100).round(2))

plt.figure(figsize=(8, 6))
plt.pie(category_counts.values, labels=category_counts.index,
autopct='%1.1f%%',
        colors=['red', 'green'], startangle=90)
plt.title('Open vs Closed Complaints')
plt.tight_layout()
plt.savefig('4_open_vs_closed.png')
plt.show()



TASK 4: Open vs Closed Status
Status_Category
Closed    1707
Open       517
Name: count, dtype: int64

Percentage:
Status_Category
Closed    76.75
Open      23.25
Name: count, dtype: float64
```



```
# TASK 5: State-wise Analysis
print('\nTASK 5: State-wise Status')

state_status = pd.crosstab(df['State'], df['Status_Category'])
state_status['Total'] = state_status.sum(axis=1)
state_status['Unresolved_Pct'] = (state_status['Open'] /
state_status['Total'] * 100).round(2)

top_states = state_status.nlargest(15, 'Total')

print('\nTop 15 States:')
print(top_states[['Open', 'Closed', 'Total',
'Unresolved_Pct']].to_string())

print('\nState with maximum complaints:',
state_status['Total'].idxmax(), '(', state_status['Total'].max(), ')')
print('State with highest unresolved rate:',
state_status['Unresolved_Pct'].idxmax(), '(',
state_status['Unresolved_Pct'].max(), '%)')

plt.figure(figsize=(14, 8))
top_states[['Closed', 'Open']].plot(kind='bar', stacked=True,
color=['green', 'red'])
plt.title('State-wise Complaint Status (Top 15 States)')
plt.xlabel('State')
plt.ylabel('Number of Complaints')
plt.legend(['Closed', 'Open'])
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig('5_state_wise_status.png')
plt.show()
```

```
TASK 5: State-wise Status

Top 15 States:
Status_Category  Open  Closed  Total  Unresolved_Pct
State
Georgia            80     208    288           27.78
Florida            39     201    240           16.25
California         61     159    220           27.73
Illinois           29     135    164           17.68
Tennessee          47      96    143           32.87
Pennsylvania       20     110    130           15.38
Michigan           23      92    115           20.00
Washington         23      75     98           23.47
Colorado           22      58     80           27.50
Maryland           15      63     78           19.23
New Jersey         19      56     75           25.33
```

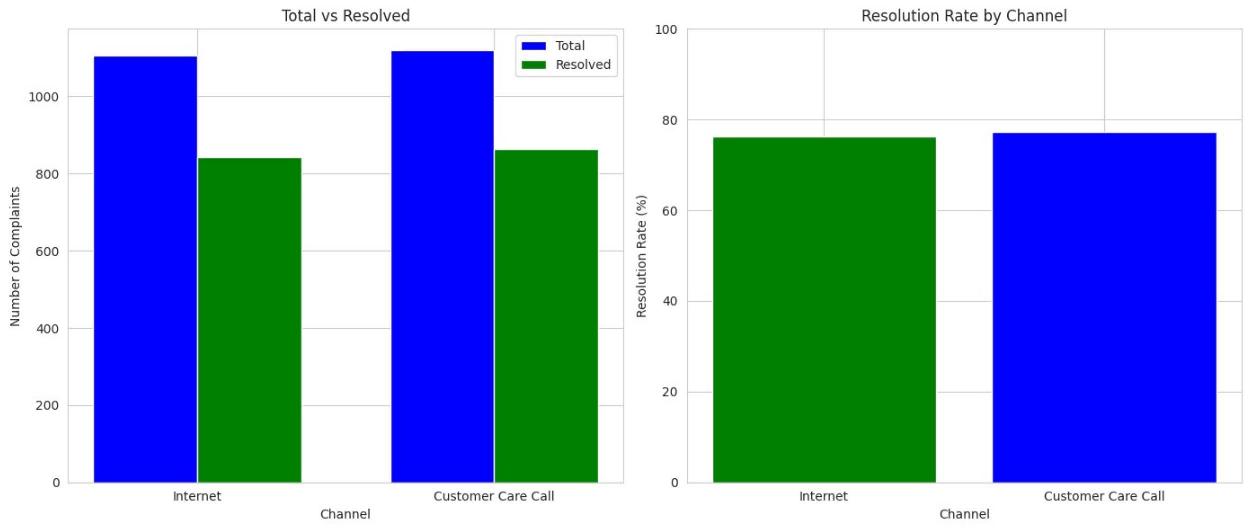
```
Texas                   22      49     71             30.99
Massachusetts           11      50     61             18.03
Virginia                11      49     60             18.33
Indiana                  9      50     59             15.25

State with maximum complaints: Georgia ( 288 )
State with highest unresolved rate: Kansas ( 50.0 %)

<Figure size 1400x800 with 0 Axes>
```



```python
# TASK 6: Channel Resolution Rate
print('\nTASK 6: Resolution Rate by Channel')

channel_stats = pd.DataFrame()
for channel in ['Internet', 'Customer Care Call']:
    channel_data = df[df['Received Via'] == channel]
    total = len(channel_data)
    resolved = len(channel_data[channel_data['Status_Category'] ==
'Closed'])
    resolution_rate = (resolved / total * 100) if total > 0 else 0

    channel_stats = pd.concat([channel_stats, pd.DataFrame({
        'Channel': [channel],
        'Total': [total],
        'Resolved': [resolved],
        'Resolution_Rate': [round(resolution_rate, 2)]
    })], ignore_index=True)

print(channel_stats.to_string(index=False))

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))
```

```python
x = range(len(channel_stats))
width = 0.35
ax1.bar([i - width/2 for i in x], channel_stats['Total'], width,
label='Total', color='blue')
ax1.bar([i + width/2 for i in x], channel_stats['Resolved'], width,
label='Resolved', color='green')
ax1.set_xlabel('Channel')
ax1.set_ylabel('Number of Complaints')
ax1.set_title('Total vs Resolved')
ax1.set_xticks(x)
ax1.set_xticklabels(channel_stats['Channel'])
ax1.legend()

ax2.bar(channel_stats['Channel'], channel_stats['Resolution_Rate'],
color=['green', 'blue'])
ax2.set_xlabel('Channel')
ax2.set_ylabel('Resolution Rate (%)')
ax2.set_title('Resolution Rate by Channel')
ax2.set_ylim(0, 100)

plt.tight_layout()
plt.savefig('6_channel_resolution.png')
plt.show()
```

```
TASK 6: Resolution Rate by Channel
           Channel  Total  Resolved  Resolution_Rate
          Internet   1105       843            76.29
Customer Care Call   1119       864            77.21
```



```python
# Summary
print('\nSUMMARY:')
print('1. Peak complaints:', monthly_complaints.idxmax(), 'with',
monthly_complaints.max(), 'complaints')
print('2. Resolved:', category_counts.get('Closed', 0), '(' +
str(round(category_counts.get('Closed', 0)/len(df)*100, 1)) + '%)')
print('3. Maximum complaint type:', complaint_types.index[0], '(' +
str(complaint_types.values[0]) + ' complaints)')
print('4. State with most complaints:',
state_status['Total'].idxmax(), '(' + str(state_status['Total'].max())
+ ')')
print('5. Internet resolution rate:',
str(channel_stats[channel_stats['Channel']=='Internet']
['Resolution_Rate'].values[0]) + '%')
print('6. Customer Care resolution rate:',
str(channel_stats[channel_stats['Channel']=='Customer Care Call']
['Resolution_Rate'].values[0]) + '%')

print('\nAnalysis Complete! 6 charts saved.')
```

```
SUMMARY:
1. Peak complaints: 2015-06 with 1046 complaints
2. Resolved: 1707 (76.8%)
3. Maximum complaint type: Internet/Speed Issues (853 complaints)
4. State with most complaints: Georgia (288)
5. Internet resolution rate: 76.29%
6. Customer Care resolution rate: 77.21%

Analysis Complete! 6 charts saved.
```