In [1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import mysql.connector
```

In [18]:
```python
db = mysql.connector.connect(host = "localhost",
                             username = "root",
                             password = "4336",
                             database = "car_theft")

cur = db.cursor()
```

In [3]:
```python
db
```

Out[3]: `<mysql.connector.connection.MySQLConnection at 0x2b7899a9e90>`

How many vehicles were stolen in total?

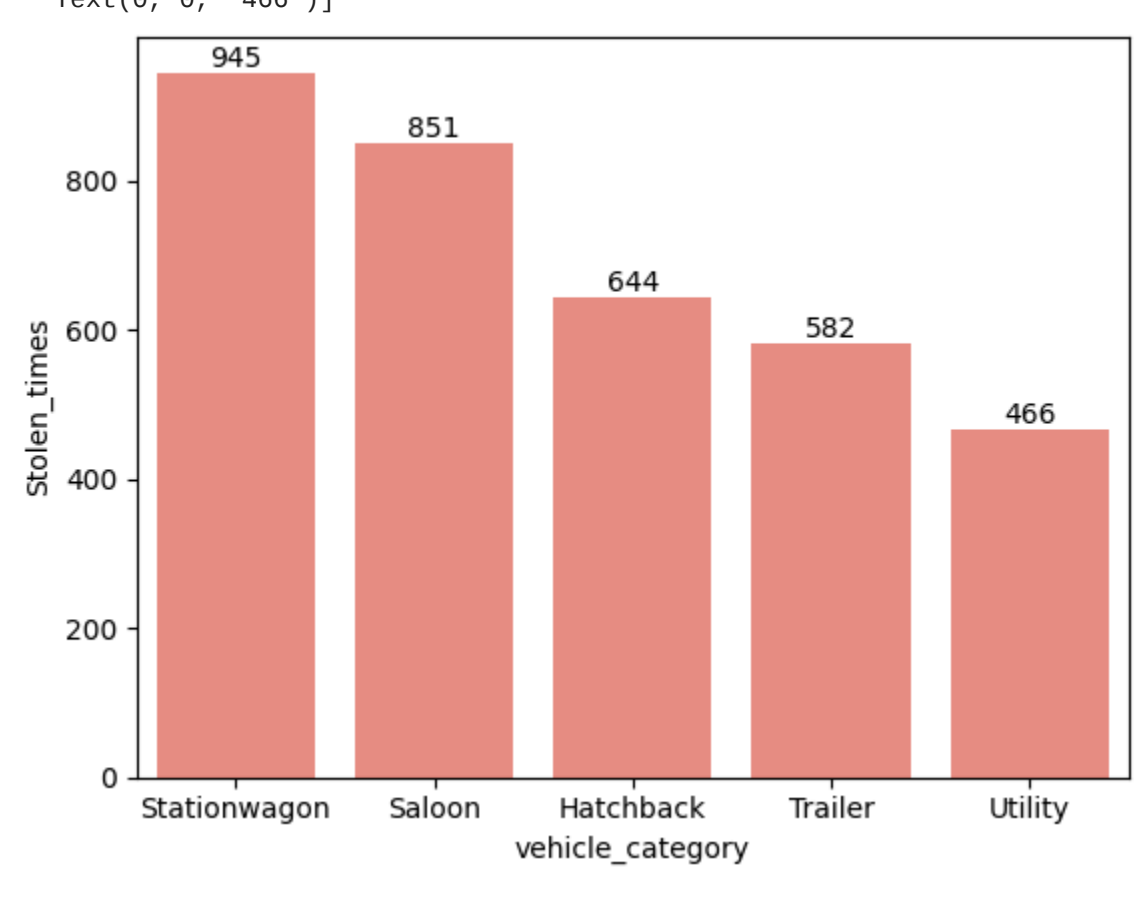
In [4]:
```python
query = """ select count("vehicle_id"
)   from stolen_vehicles
 """

cur.execute(query)

data = cur.fetchall()

print("Numbers of vehicles stolen --", data[0][0])
```

```
Numbers of vehicles stolen -- 4538
```

What are the top 5 most commonly stolen vehicle types

In [5]:
```python
query =""" select vehicle_type, count("vehicle_id") as  theft_count from stolen_vehicles
group by vehicle_type
ORDER BY theft_count DESC
LIMIT 5;
 """

cur.execute(query)


data = cur.fetchall()

df = pd.DataFrame(data, columns = ["vehicle_category", "Stolen_times"])
ax = sns.barplot(x= df["vehicle_category"] ,y =  df["Stolen_times"], data = df, color = "salmon")
plt.xticks(rotation = 0)
ax.bar_label(ax.containers[0])
```

Out[5]:
```
[Text(0, 0, '945'),
 Text(0, 0, '851'),
 Text(0, 0, '644'),
 Text(0, 0, '582'),
 Text(0, 0, '466')]
```



What is the most common color of stolen vehicles?

In [6]:
```python
query =""" select color, count("vehicle_id") as  Stolen_times from stolen_vehicles
group by color
ORDER BY Stolen_times DESC
;
 """

cur.execute(query)


data = cur.fetchall()
print(data[0])
```

```
('Silver', 1272)
```

Which region has reported the most vehicle thefts?

In [7]:
```python
query =""" select l.region , count("vehicle_id") as theft_vehicle
from locations as l
inner join stolen_vehicles as s
on l.ï»¿location_id = s.location_id
group by l.region
order by theft_vehicle desc
;
 """

cur.execute(query)


data = cur.fetchall()
df = pd.DataFrame(data, columns = ["Region", "Stolen_times"])
df
```

Out[7]:

	Region	Stolen_times
0	Auckland	1630
1	Canterbury	660
2	Bay of Plenty	445
3	Wellington	417
4	Waikato	369
5	Northland	234
6	Gisborne	175
7	Otago	139
8	ManawatÅ«-Whanganui	139
9	Taranaki	112
10	Hawke's Bay	100
11	Nelson	92
12	Southland	26

What is the average age of stolen vehicles (based on model year)?

In [8]:
```python
query =""" SELECT AVG("model_year"
) as avg_age
FROM stolen_vehicles;
 """

cur.execute(query)


data = cur.fetchall()
print(data)
```

```
[(0.0,)]
```

What is the trend of vehicle thefts over the years? Is it increasing or decreasing?

In [9]:
```python
query = """
SELECT YEAR(STR_TO_DATE(date_stolen, '%m/%d/%Y')) AS year, COUNT(*) AS theft_count
FROM stolen_vehicles
GROUP BY YEAR(STR_TO_DATE(date_stolen, '%m/%d/%Y'))
"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns=['year', 'theft_count'])
df
```

Out[9]:

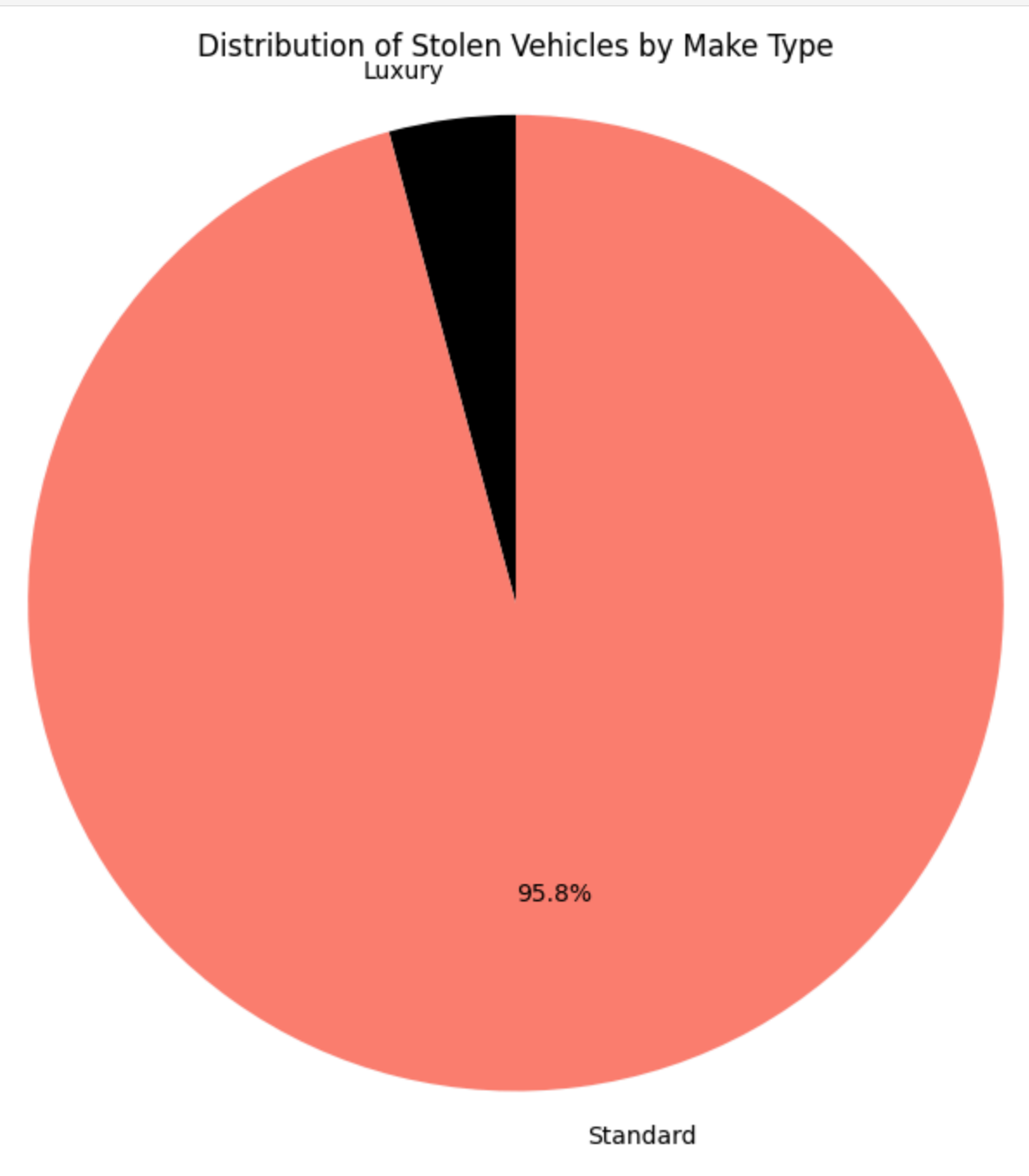
	year	theft_count
0	2021	1663
1	2022	2875

What percentage of stolen vehicles are luxury makes versus standard makes?

In [10]:
```python
query = """
SELECT
    ROUND(SUM(CASE WHEN md.make_type = 'Luxury' THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 2) as luxury_percentage,
    ROUND(SUM(CASE WHEN md.make_type = 'Standard' THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 2) as standard_percentage
FROM
    stolen_vehicles sv
JOIN
    make_details md ON sv.make_id = md.ï»¿make_id;

"""
cur.execute(query)
data = cur.fetchall()


# Extract the percentages from the data
luxury_percentage = data[0][0]
standard_percentage = data[0][1]

print(luxury_percentage)
print(standard_percentage)
```

```
4.19
95.81
```

In [11]:
```python
labels = ['Luxury', 'Standard']
sizes = [luxury_percentage, standard_percentage]

# Create the pie chart
plt.figure(figsize=(8, 8))
colors = ('black','salmon')
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90,colors=colors)
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle

# Add a title
plt.title('Distribution of Stolen Vehicles by Make Type')

# Show the plot
plt.show()
```



Which combination of vehicle type, make, and color is most frequently stolen?

In [12]:
```python
query =""" SELECT sv.vehicle_type, md.make_name, sv.color, COUNT(*) as theft_count
FROM stolen_vehicles sv
JOIN make_details md ON sv.make_id = md.ï»¿make_id
GROUP BY sv.vehicle_type, md.make_name, sv.color
ORDER BY theft_count DESC
LIMIT 1;
 """

cur.execute(query)


data = cur.fetchall()
df = pd.DataFrame(data, columns=['vehicle_type', 'make_name','color', 'theft_count'])
df
```

Out[12]:

	vehicle_type	make_name	color	theft_count
0	Trailer	Trailer	Silver	259

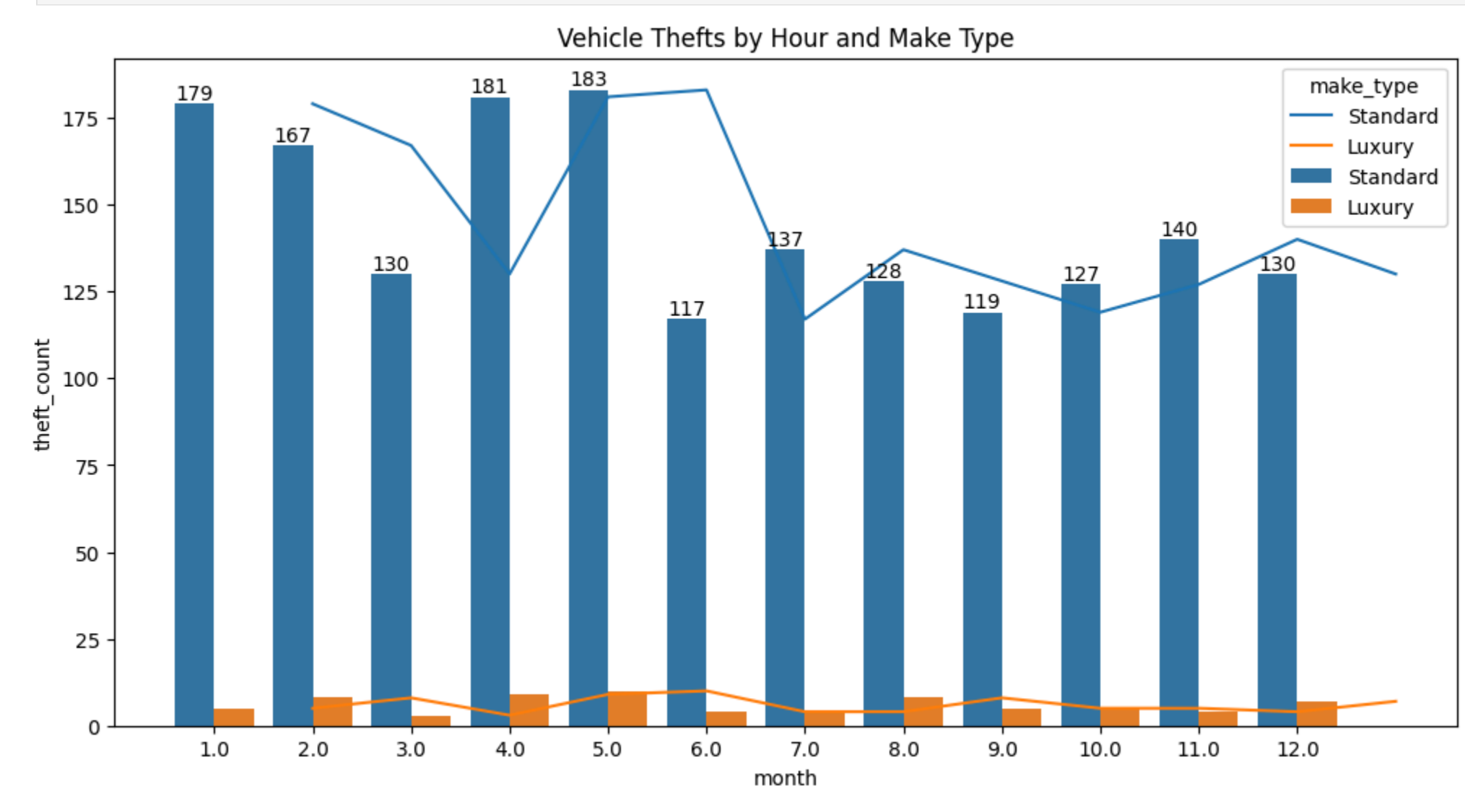
Can we identify any patterns in the timing and location of luxury vehicle thefts compared to standard vehicle thefts?

In [13]:
```python
query =""" SELECT
    md.make_type,
    EXTRACT(MONTH FROM sv.date_stolen) AS month,
    COUNT(*) AS theft_count
FROM
    stolen_vehicles sv
JOIN
    make_details md ON sv.make_id = md.ï»¿make_id
GROUP BY
    md.make_type, month
ORDER BY
  theft_count DESC;
 """

cur.execute(query)


data = cur.fetchall()
df = pd.DataFrame(data, columns=['make_type','month', 'theft_count'])
df
```

Out[13]:

	make_type	month	theft_count
0	Standard	NaN	2610
1	Standard	5.0	183
2	Standard	4.0	181
3	Standard	1.0	179
4	Standard	2.0	167
5	Standard	11.0	140
6	Standard	7.0	137
7	Standard	12.0	130
8	Standard	3.0	130
9	Standard	8.0	128
10	Standard	10.0	127
11	Standard	9.0	119
12	Luxury	NaN	118
13	Standard	6.0	117
14	Luxury	5.0	10
15	Luxury	4.0	9
16	Luxury	2.0	8
17	Luxury	8.0	8
18	Luxury	12.0	7
19	Luxury	10.0	5
20	Luxury	9.0	5
21	Luxury	1.0	5
22	Luxury	7.0	4
23	Luxury	6.0	4
24	Luxury	11.0	4
25	Luxury	3.0	3

In [14]:
```python
plt.figure(figsize=(12, 6))
bx=sns.barplot(x='month', y='theft_count',hue='make_type', data=df)
bx.bar_label(bx.containers[0])
sns.lineplot(x='month', y='theft_count',hue='make_type', data=df)

plt.title('Vehicle Thefts by Hour and Make Type')
plt.show()
```



In [15]:
```python
query =""" SELECT
    DATE(sv.date_stolen) as ds,
    l.region,
    COUNT(*) as y
FROM
    stolen_vehicles sv
JOIN
    locations l ON sv.location_id = l.ï»¿location_id
GROUP BY
    DATE(sv.date_stolen), l.region
ORDER BY
    l.region, DATE(sv.date_stolen)
 """

cur.execute(query)


data = cur.fetchall()
df = pd.DataFrame(data, columns=['stolen_date','region', 'theft_count'])
df
```

Out[15]:

	stolen_date	region	theft_count
0	None	Auckland	964
1	0001-01-22	Auckland	7
2	0001-02-22	Auckland	5
3	0001-03-22	Auckland	4
4	0001-04-22	Auckland	9
...	...	...	...
591	2012-08-21	Wellington	1
592	2012-09-21	Wellington	2
593	2012-10-21	Wellington	1
594	2012-11-21	Wellington	4
595	2012-12-21	Wellington	3

596 rows × 3 columns

In [19]:
```python
import mysql.connector
import pandas as pd
import numpy as np
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
from math import sqrt
import matplotlib.pyplot as plt

cursor = db.cursor()

# SQL query to get the number of thefts by region and date
query = """
SELECT l.region, sv.date_stolen, COUNT(*) as theft_count
FROM stolen_vehicles sv
JOIN locations l ON sv.location_id = l.ï»¿location_id
GROUP BY l.region, sv.date_stolen
ORDER BY l.region, sv.date_stolen
"""

cursor.execute(query)
result = cursor.fetchall()

# Convert the result to a DataFrame
df = pd.DataFrame(result, columns=['region', 'date', 'theft_count'])
df['date'] = pd.to_datetime(df['date'])

# Function to forecast thefts for a region
def forecast_thefts(region_data):
    # Prepare the data
    region_data = region_data.set_index('date')
    region_data = region_data.resample('D').sum().fillna(0)

    # Fit ARIMA model
    model = ARIMA(region_data['theft_count'], order=(1,1,1))
    results = model.fit()

    # Forecast for the next year
    forecast = results.forecast(steps=365)
    return forecast

# Dictionary to store forecasts
forecasts = {}

# Perform forecasting for each region
for region in df['region'].unique():
    region_data = df[df['region'] == region]
    forecast = forecast_thefts(region_data)
    forecasts[region] = forecast

# Print the forecasted number of thefts for each region
for region, forecast in forecasts.items():
    print(f"Forecasted thefts for {region} in the next year: {int(forecast.sum())}")
```

```
C:\Users\nikde\AppData\Local\Temp\ipykernel_23524\3105637162.py:25: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please s
pecify a format.
  df['date'] = pd.to_datetime(df['date'])
C:\Users\nikde\AppData\Roaming\Python\Python311\site-packages\statsmodels\tsa\statespace\sarimax.py:978: UserWarning: Non-invertible starting MA parameters found. Using zeros as starting parameters.
  warn('Non-invertible starting MA parameters found.'
Forecasted thefts for Auckland in the next year: 8096
Forecasted thefts for Bay of Plenty in the next year: 946
Forecasted thefts for Canterbury in the next year: 1676
Forecasted thefts for Gisborne in the next year: 433
Forecasted thefts for Hawke's Bay in the next year: 302
Forecasted thefts for ManawatÅ«-Whanganui in the next year: 393
Forecasted thefts for Nelson in the next year: 305
Forecasted thefts for Northland in the next year: 643
Forecasted thefts for Otago in the next year: 540
Forecasted thefts for Southland in the next year: 59
Forecasted thefts for Taranaki in the next year: 343
Forecasted thefts for Waikato in the next year: 1351
Forecasted thefts for Wellington in the next year: 1583
```

The current forecast suggests that Auckland has the highest predicted number of thefts (8096), followed by Canterbury (1676) and Wellington (1583). However, given the straight-line nature of the predictions, these forecasts may not be capturing the full complexity of theft patterns in these regions.