Predictive analysis of naval incidents in the USA, 2002 - 2015:

Annex 4.1. Data Explore: VesselBalancedSample

Author: Oscar Anton
Date: 2024

Version: 0.9

0. Loadings

Libraries

In [3]:
```
# Data general management

import pandas as pd

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Automatic Exploratory Data Analysis (EDA) report
from ydata_profiling import ProfileReport

# Ignore warnings
import warnings
warnings.filterwarnings("ignore")
```

General variables

In [4]:
```
# Main data folder
merged_activity_folder = '../3.DataPreprocess/DataMergedActivity'

# Toggle for export data to external file
file_export_enabled = False
```

Load base dataframe

In [5]:
```
# Load dataframe from external file
VesselBalancedSample = pd.read_feather(merged_activity_folder + '/' + 'VesselBalancedSam

# Check dataframe structure
print(f'VesselBalancedSample {VesselBalancedSample.shape} loaded')
VesselBalancedSample.head()
```

```
VesselBalancedSample (109836, 13) loaded
```

Out[5]:

	index	vessel_id	imo_number	vessel_name	vessel_class	build_year	gross_ton	length	flag_
0	0	5820		ISABELLA MARIE	Recreational	1999	14	32.8	
1	1	170582		TERMINATOR	Fishing Vessel	1979	17	38.1	
2	2	257931		SUMMER ISLE	Recreational	1984	8	29.9	
3	3	151752		NORJERNAN	Passenger Ship	1976	18	35.2	
4	4	308953		NONSENSE	Recreational	1987	8	27.0	

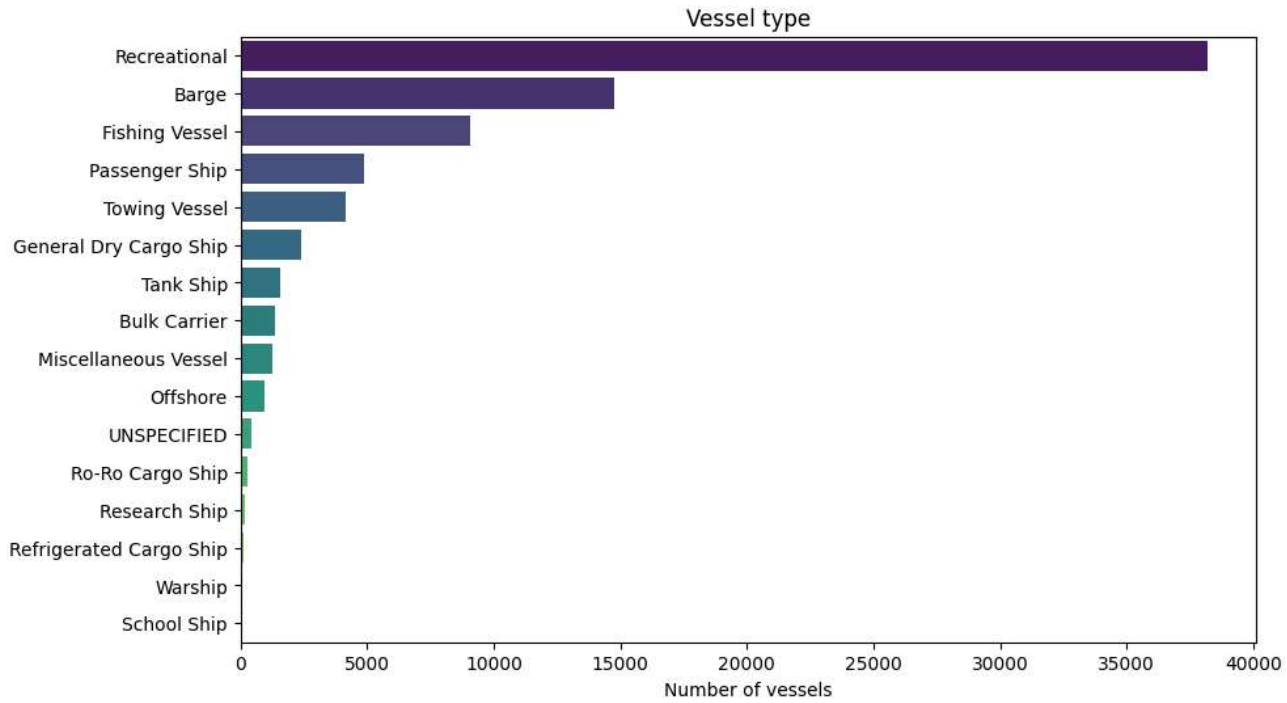
1. Vessel features

1.1. vessel_class

Frequency

In [6]:

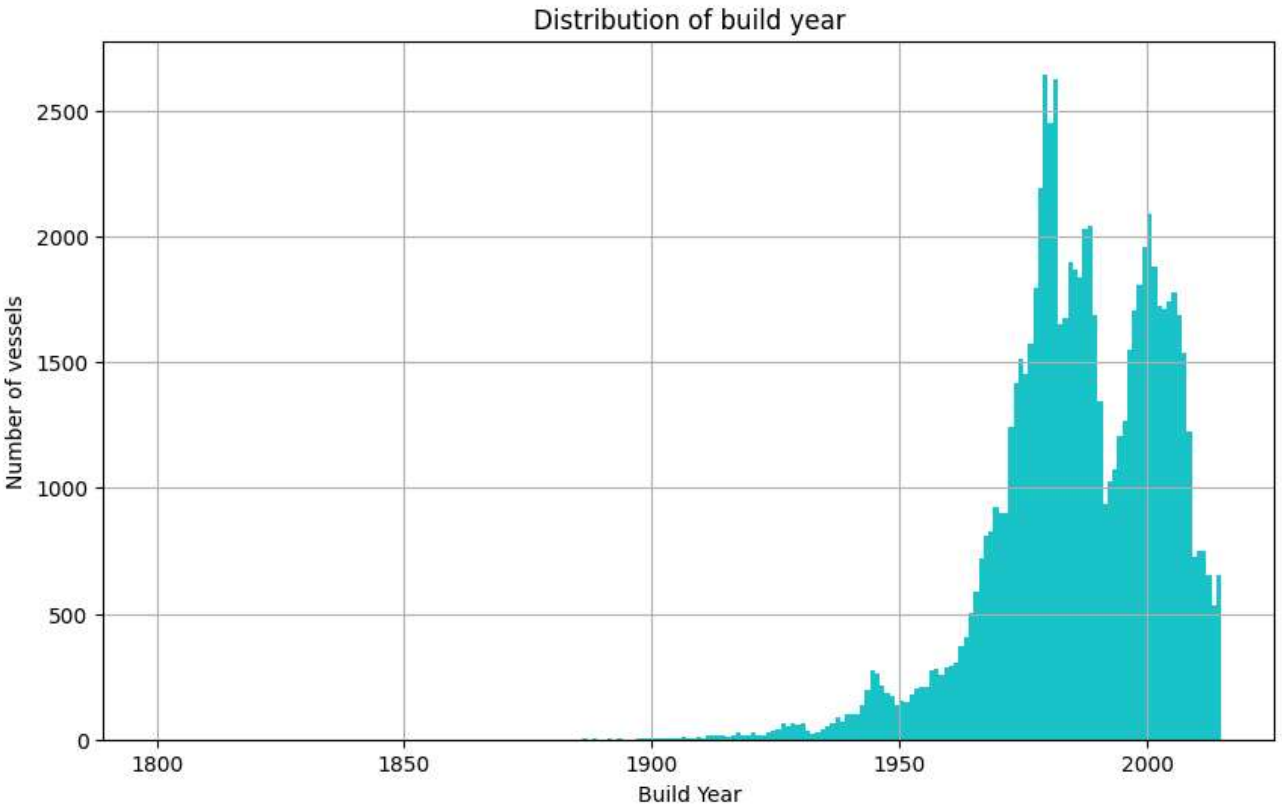
```
# Filter data: group by vessel class
filtered_df = (VesselBalancedSample
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('vessel_class').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=False))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='frequency', y='vessel_class', data=filtered_df, palette='viridis')

# Customize plot
plt.title('Vessel type')
plt.xlabel('Number of vessels')
plt.ylabel(None)
plt.show()
```



1.2. build_year

Frequency

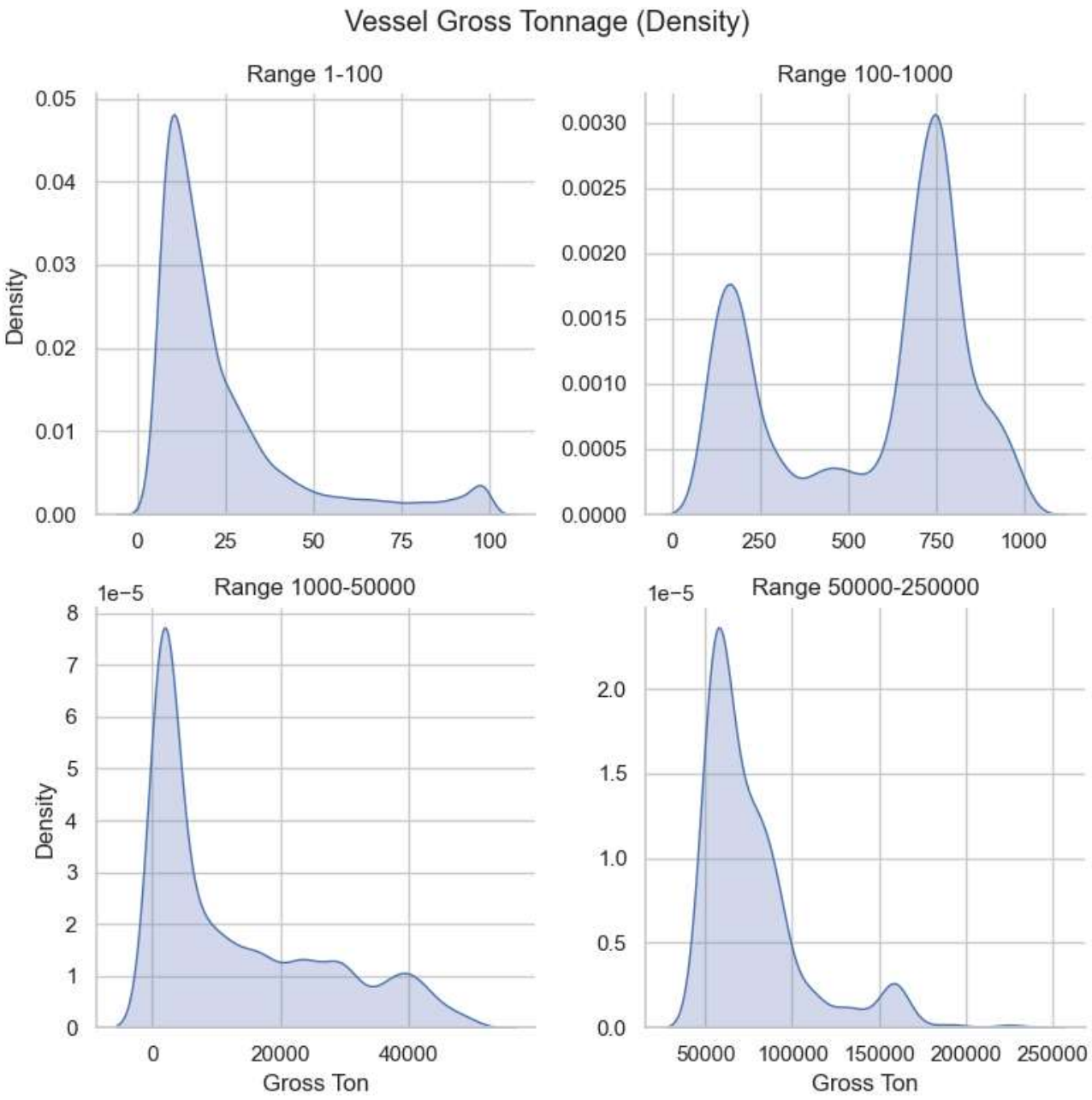
In [7]:
```
# Filter data: build_year between 1800 and 2015
filtered_df = (VesselBalancedSample
               [(VesselBalancedSample['build_year'] >= 1800) & (VesselBalancedSample['bu
               .drop_duplicates(subset='vessel_id', keep='first'))

# Plot histogram
plt.figure(figsize=(10, 6))
sns.histplot(filtered_df['build_year'], bins=range(1800, 2016), edgecolor='None', color=

# Customize plot
plt.title('Distribution of build year')
plt.xlabel('Build Year')
plt.ylabel('Number of vessels')
plt.grid(True)
plt.show()
```



1.3. gross_tonnage

Density

In [8]:
```
# Filter data
filtered_df = (VesselBalancedSample
               [(VesselBalancedSample['gross_ton'] >= 1) & (VesselBalancedSample['gross_
               .drop_duplicates(subset='vessel_id', keep='first'))

# Labels for facet wrap
labels = ["1-100", "100-1000", "1000-50000", "50000-250000"]
filtered_df['gross_ton_range'] = pd.cut(filtered_df['gross_ton'], bins=[0, 100, 1000, 50

# Plot
sns.set(style="whitegrid")
g = sns.FacetGrid(filtered_df, col="gross_ton_range", col_wrap=2, height=4, sharey=False
g.map(sns.kdeplot, "gross_ton", shade=True)
g.set_axis_labels("Gross Ton", "Density")
g.set_titles("Range {col_name}")

# Customize plot
plt.suptitle('Vessel Gross Tonnage (Density)')
plt.tight_layout()
plt.show()
```



Ranking

In [9]:
```
# Filter data
filtered_df = (VesselBalancedSample
               [['vessel_id', 'imo_number', 'vessel_name', 'build_year', 'gross_ton', 'l
               .sort_values(by='gross_ton', ascending=False)
               .drop_duplicates()
               .head(10))

# Table output
print(filtered_df)
```

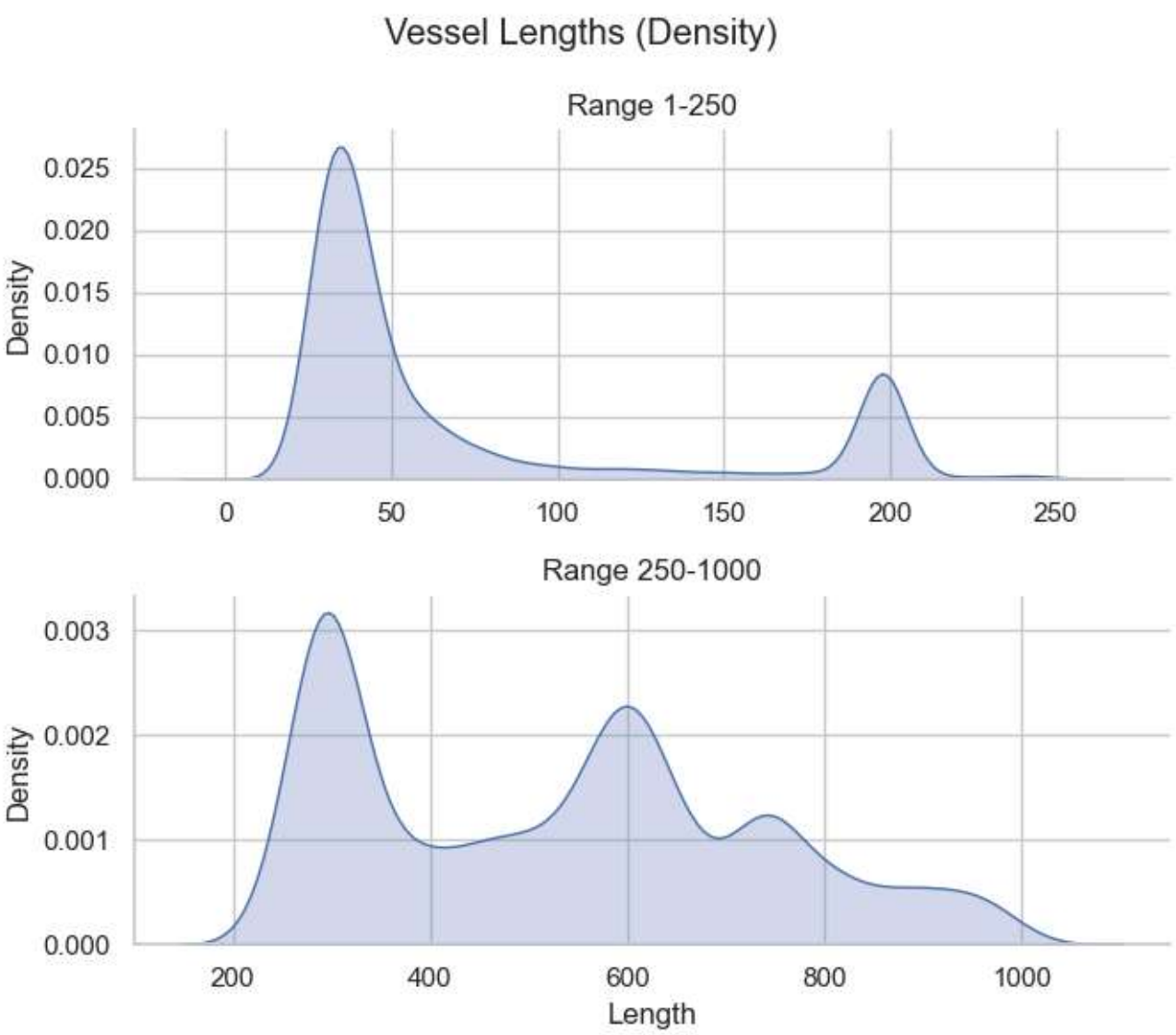
```
        vessel_id imo_number            vessel_name  build_year  gross_ton  \
34046      299897    7376525        KAPETAN GIANNIS        1977     234627
101554     933484    9383948     ALLURE OF THE SEAS        2010     225282
92389      933483    9383936      OASIS OF THE SEAS        2009     225282
21540      324613    7370301     KAPETAN PANAGIOTIS        1977     218447
31454      224539    7376989  CHEVRON SOUTH AMERICA        1976     198951
45688      260729    7373298                AURIGA        1976     194992
1671       228357    7708302             FOLK MOON        1981     188728
55743      228358    7708314          BERGE PIONEER        1980     188728
46662      881546    9266102                YM SKY        2003     179037
42549      275226    7389534          BERGE INGERID        1977     169752

        length
34046    406.6
101554  1181.0
92389   1187.0
21540    362.3
31454   1200.4
45688    378.0
1671    1117.0
55743   1071.7
46662    172.0
42549    362.6
```

1.4. Vessel length

Densitity

In [10]:

```
# Filter data
filtered_df = (VesselBalancedSample
               [(VesselBalancedSample['length'] >= 1) & (VesselBalancedSample['length']
               .drop_duplicates(subset='vessel_id', keep='first'))

# Labels for facet wrap
labels = ["1-250", "250-1000"]
filtered_df['length_range'] = pd.cut(filtered_df['length'], bins=[0, 250, 1000], labels=

# Plot
g = sns.FacetGrid(filtered_df, col="length_range", col_wrap=1, height=3, aspect=7/3, sha
g.map(sns.kdeplot, "length", shade=True)

# Customize plot
g.set_axis_labels("Length", "Density")
g.set_titles("Range {col_name}")

plt.suptitle('Vessel Lengths (Density)')
plt.tight_layout()
plt.show()
```



Ranking

In [11]:
```
# Filter data
filtered_df =(VesselBalancedSample
              [['vessel_id', 'imo_number', 'vessel_name', 'build_year', 'gross_ton', 'le
              .sort_values(by='length', ascending=False)
              .drop_duplicates()
              .head(10))

# Table output
print(filtered_df)
```

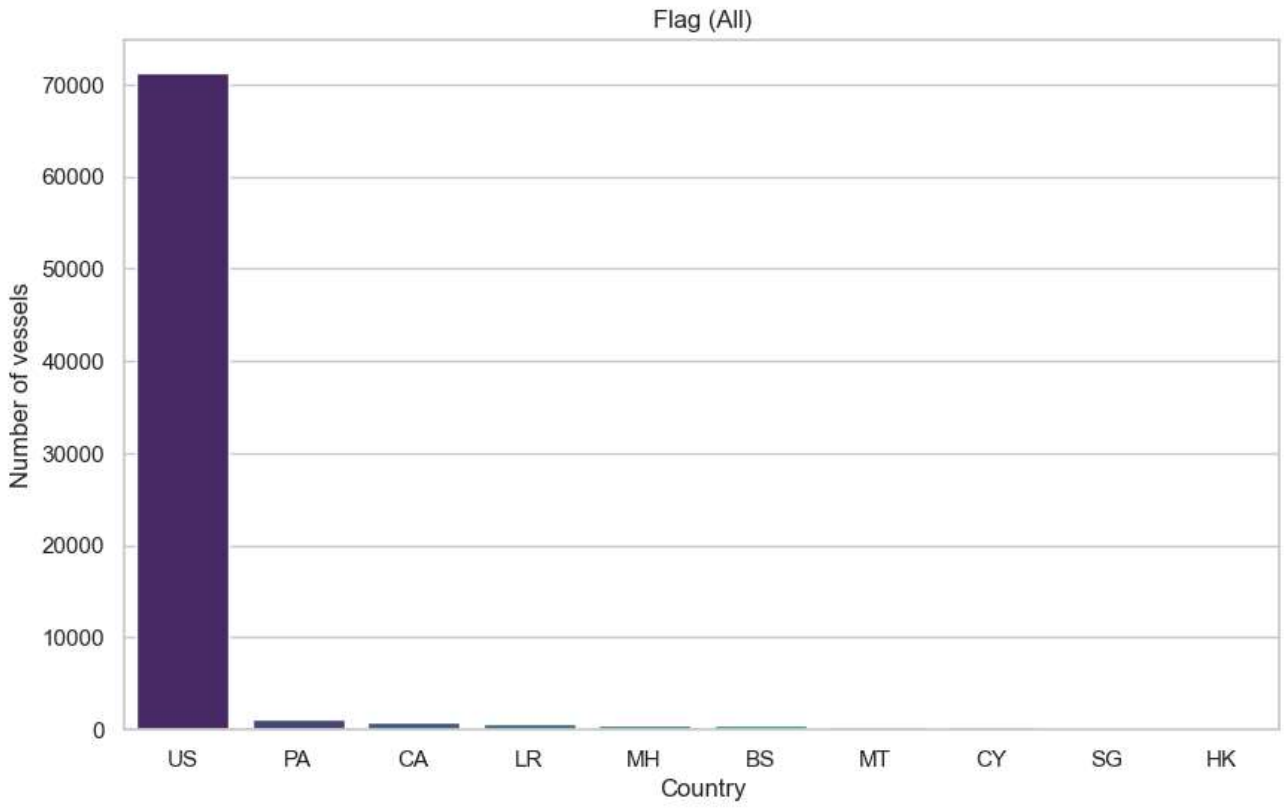
```
        vessel_id imo_number            vessel_name  build_year  gross_ton  \
88075     1001188    9302889          GRETE MAERSK        2005      97933
103054     998455    9302877         GUDRUN MAERSK        2005      97933
91605     1028411    9359052       MATHILDE MAERSK        2009      98268
94740      999387    9359014        MARCHEN MAERSK        2007      98268
89717     1008200    9359040          MARIT MAERSK        2009      98268
37272     1277844    9472127         COSCO FORTUNE        2012     141823
29921     1325557    9447902          MSC FILLIPPA        2011     140259
31454      224539    7376989  CHEVRON SOUTH AMERICA        1976     198951
19945     1171505    9398371              MSC IVANA        2008     131771
29356      274926    7359058               KAROLINE        1976     158475

        length
88075   1203.8
103054  1203.8
91605   1203.7
94740   1203.7
89717   1203.7
37272   1202.2
29921   1201.0
31454   1200.4
19945   1192.9
29356   1192.0
```

1.5. Flag

Frequency (All)

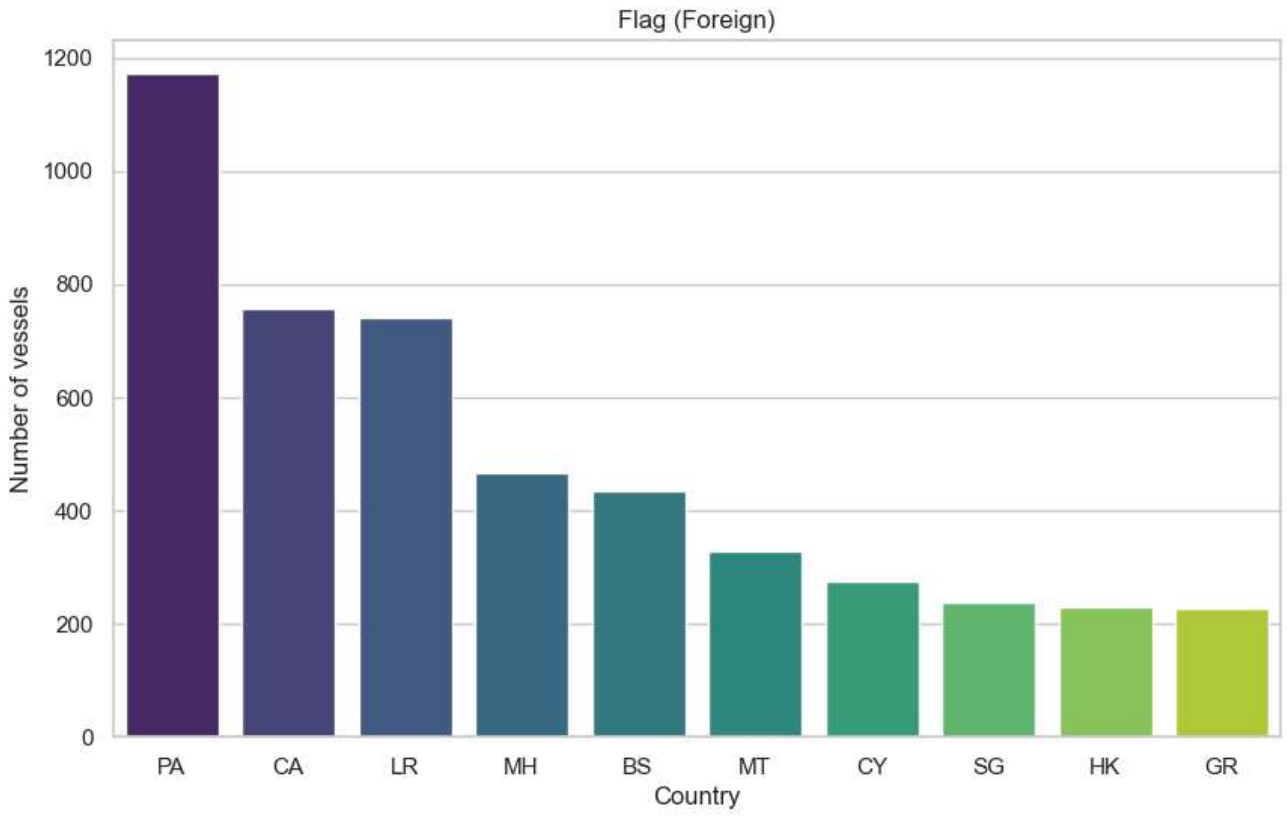
In [12]:

```python
# Filter data: group by vessel class
filtered_df = (VesselBalancedSample
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('flag_abbr').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=False)
               .head(10))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='flag_abbr', y='frequency', data=filtered_df, palette='viridis')

# Customize plot
plt.title('Flag (All)')
plt.xlabel('Country')
plt.ylabel('Number of vessels')
plt.show()
```



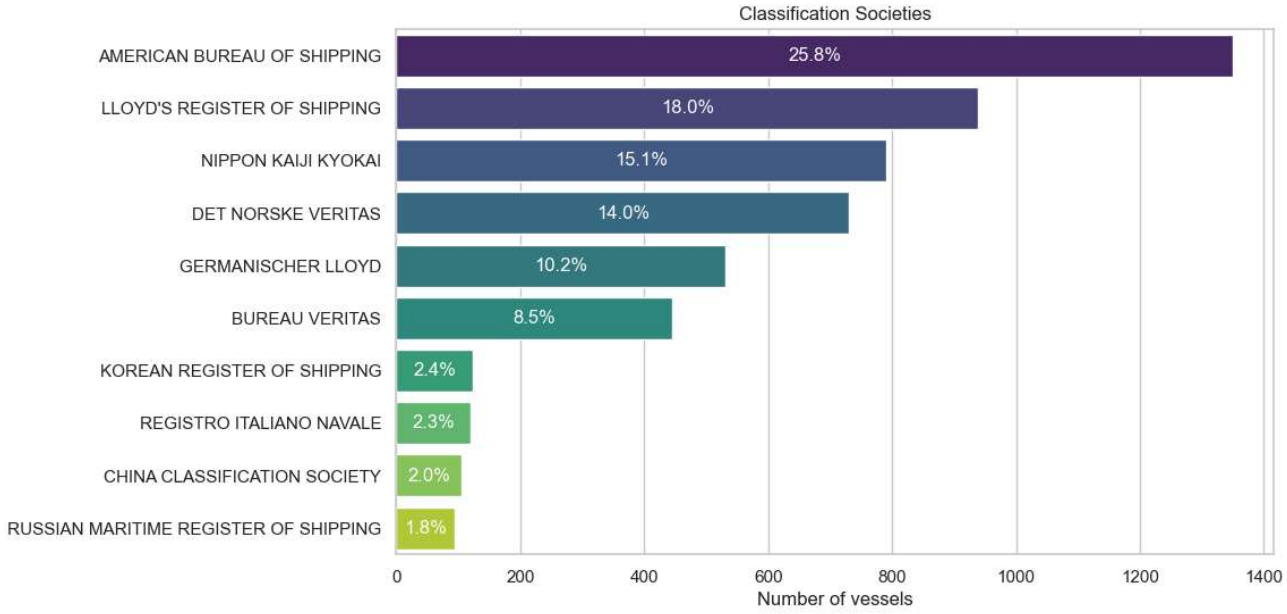
Frequency (Foreign)

In [13]:
```
# Filter data: drop US flag, group by vessel class
filtered_df = (VesselBalancedSample
               [VesselBalancedSample['flag_abbr'] != "US"]
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('flag_abbr').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=False)
               .head(10))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='flag_abbr', y='frequency', data=filtered_df, palette='viridis')

# Customize plot
plt.title('Flag (Foreign)')
plt.xlabel('Country')
plt.ylabel('Number of vessels')
plt.show()
```



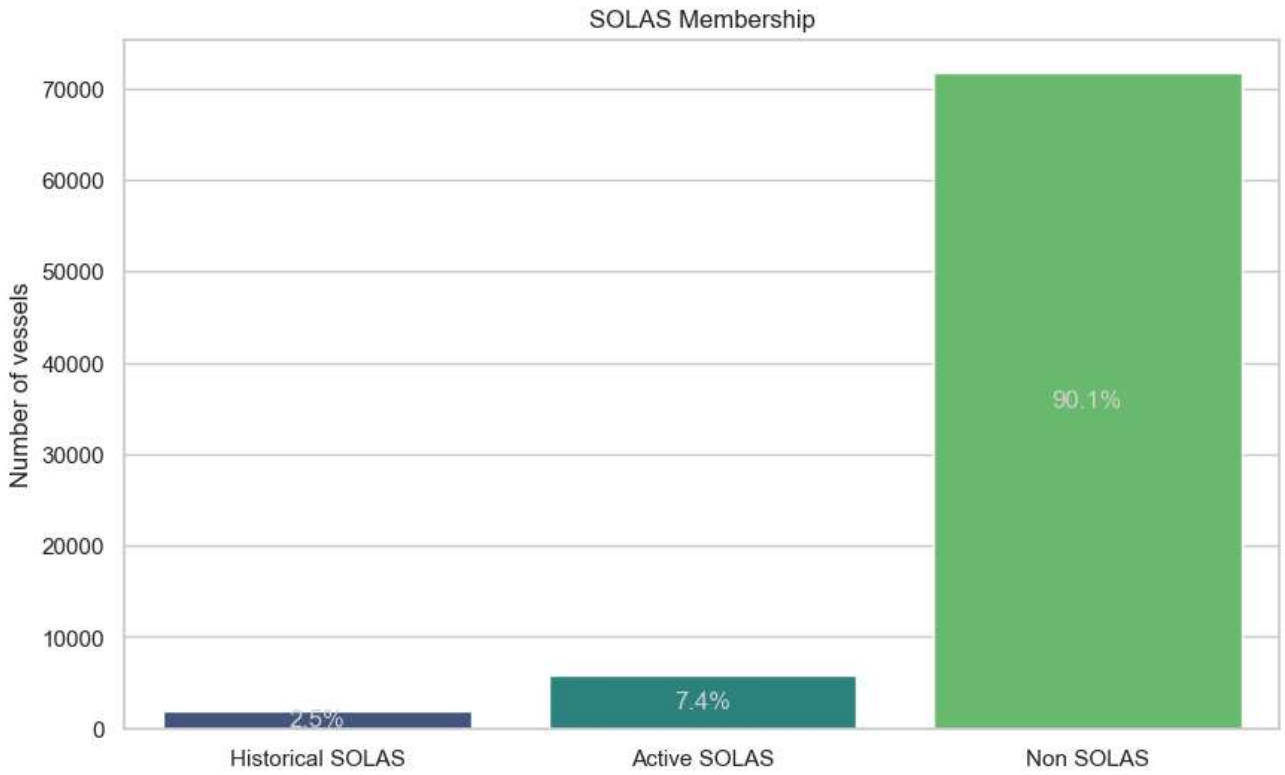
1.6. Classification Societies

In [14]:
```
# Filter data: group by classification_society
filtered_df = (VesselBalancedSample
               [VesselBalancedSample['classification_society'] != "UNSPECIFIED"]
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('classification_society').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=False)
               .head(10))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='frequency', y='classification_society', data=filtered_df, palette='viridi

# Percentage
filtered_df['percentage'] = filtered_df['frequency'] / filtered_df['frequency'].sum() *
for i, (value, percentage) in enumerate(zip(filtered_df['frequency'], filtered_df['perce
    plt.text(value / 2, i, f'{percentage:.1f}%', va='center', ha='center', color='white'

# Customize plot
plt.title('Classification Societies')
plt.xlabel('Number of vessels')
plt.ylabel(None)
plt.show()
```



1.7. SOLAS Membership

In [15]:
```
# Filter data: group by SOLAS
filtered_df = (VesselBalancedSample
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('solas_desc').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=True))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='solas_desc', y='frequency', data=filtered_df, palette='viridis')

# Percentage
filtered_df['percentage'] = filtered_df['frequency'] / filtered_df['frequency'].sum() *
for i, (value, percentage) in enumerate(zip(filtered_df['frequency'], filtered_df['perce
    plt.text(i, value / 2, f'{percentage:.1f}%', va='center', ha='center', color='lightg

# Customize plot
plt.title('SOLAS Membership')
plt.xlabel(None)
plt.ylabel('Number of vessels')
plt.show()
```

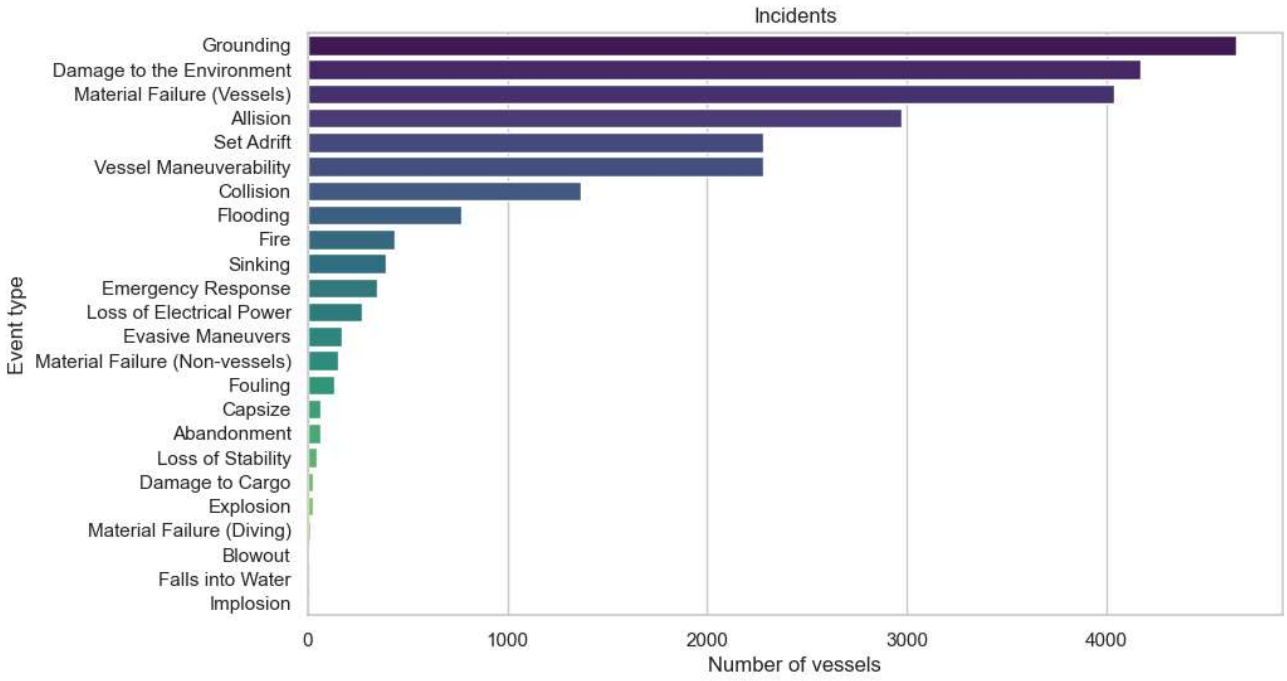


2. Incidents

2.1. event_type

Frequency

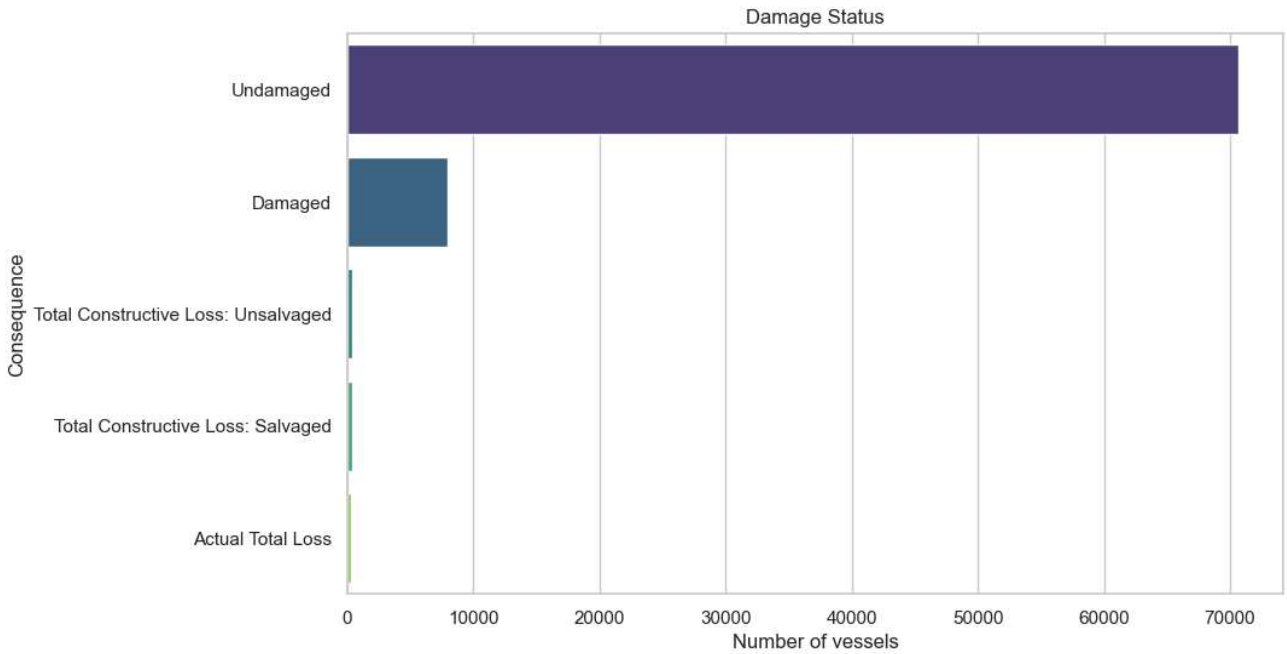
In [16]:
```
# Filter data: group by event_type
filtered_df = (VesselBalancedSample
               [VesselBalancedSample['event_type'] != "No event"]
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('event_type').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=False))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='frequency', y='event_type', data=filtered_df, palette='viridis')

# Customize plot
plt.title('Incidents')
plt.xlabel('Number of vessels')
plt.ylabel('Event type')
plt.show()
```



2.2. damage_status

Frequency

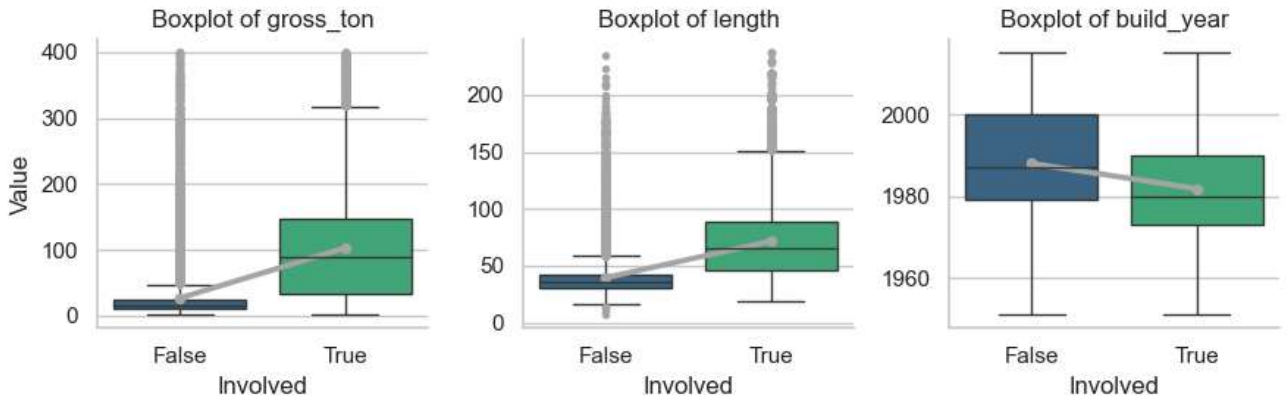
In [17]:
```
# Filter data: group by damage_status
filtered_df = (VesselBalancedSample
               .drop_duplicates(subset='vessel_id', keep='first')
               .groupby('damage_status').size().reset_index(name='frequency')
               .sort_values(by='frequency', ascending=False))

# Plot barplot
plt.figure(figsize=(10, 6))
sns.barplot(x='frequency', y='damage_status', data=filtered_df, palette='viridis')

# Customize plot
plt.title('Damage Status')
plt.xlabel('Number of vessels')
plt.ylabel('Consequence')
plt.show()
```



3. Incident Involvement

3.1. Incident Involvement / Vessel features

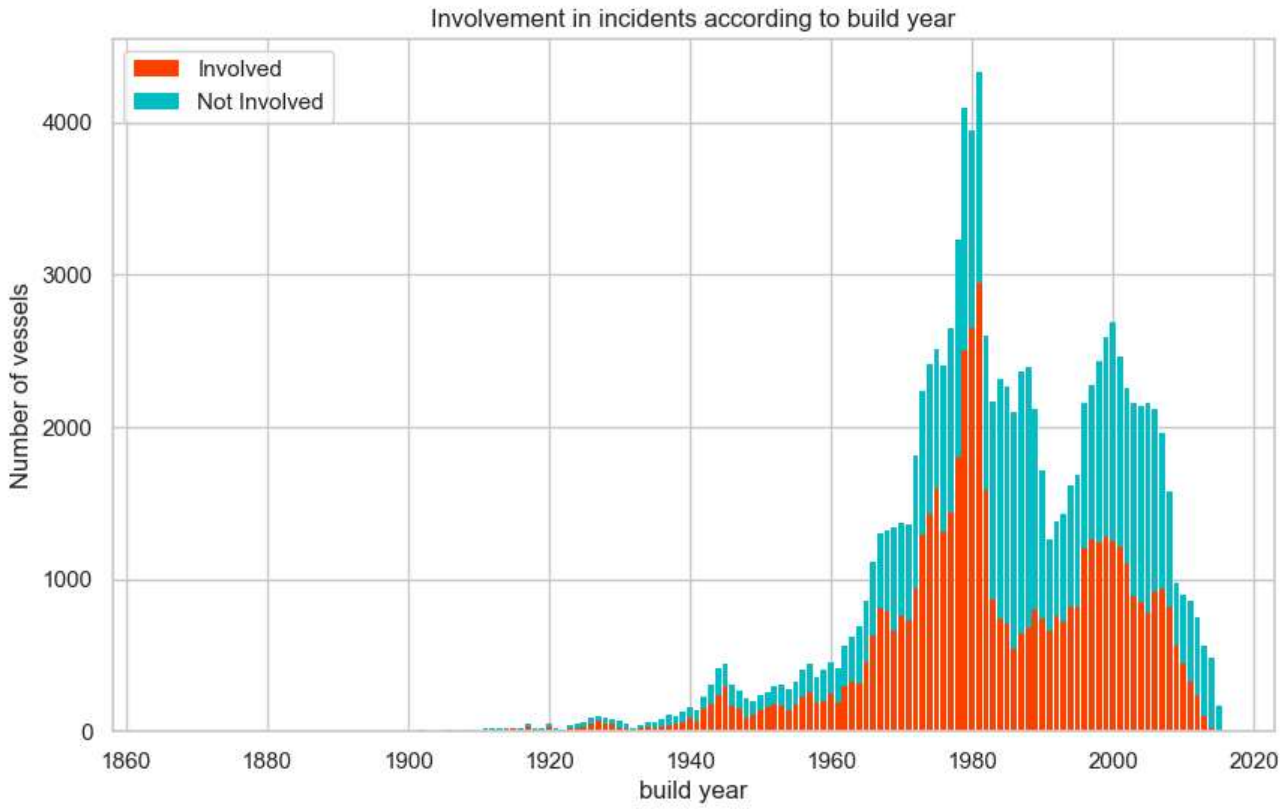
In [18]:
```
# Add 'involved' (true/false) variable to base dataframe
VesselBalancedSample['involved'] = VesselBalancedSample['event_type'] != "No event"

# Filter data
filtered_df = (VesselBalancedSample
               [['involved', 'gross_ton', 'length', 'build_year']]
               [(VesselBalancedSample['gross_ton'] < 400) &
                (VesselBalancedSample['length'] < 250) &
                (VesselBalancedSample['build_year'] > 1950)])

# Pivot data
filtered_df = pd.melt(filtered_df, id_vars='involved', var_name='variable', value_name='

# Plot boxplot
g = sns.FacetGrid(filtered_df, col='variable', sharey=False, col_wrap=3)
g.map_dataframe(sns.boxplot, x='involved', y='value', palette='viridis',
                flierprops=dict(markerfacecolor='darkgrey', markeredgecolor='none', mark
g.map_dataframe(sns.pointplot, x='involved', y='value', color='darkgrey', markers='.')

# Customize plot
g.set_titles("Boxplot of {col_name}")
g.set_axis_labels("Involved", "Value")

plt.show()
```



Incident involvement / build_year

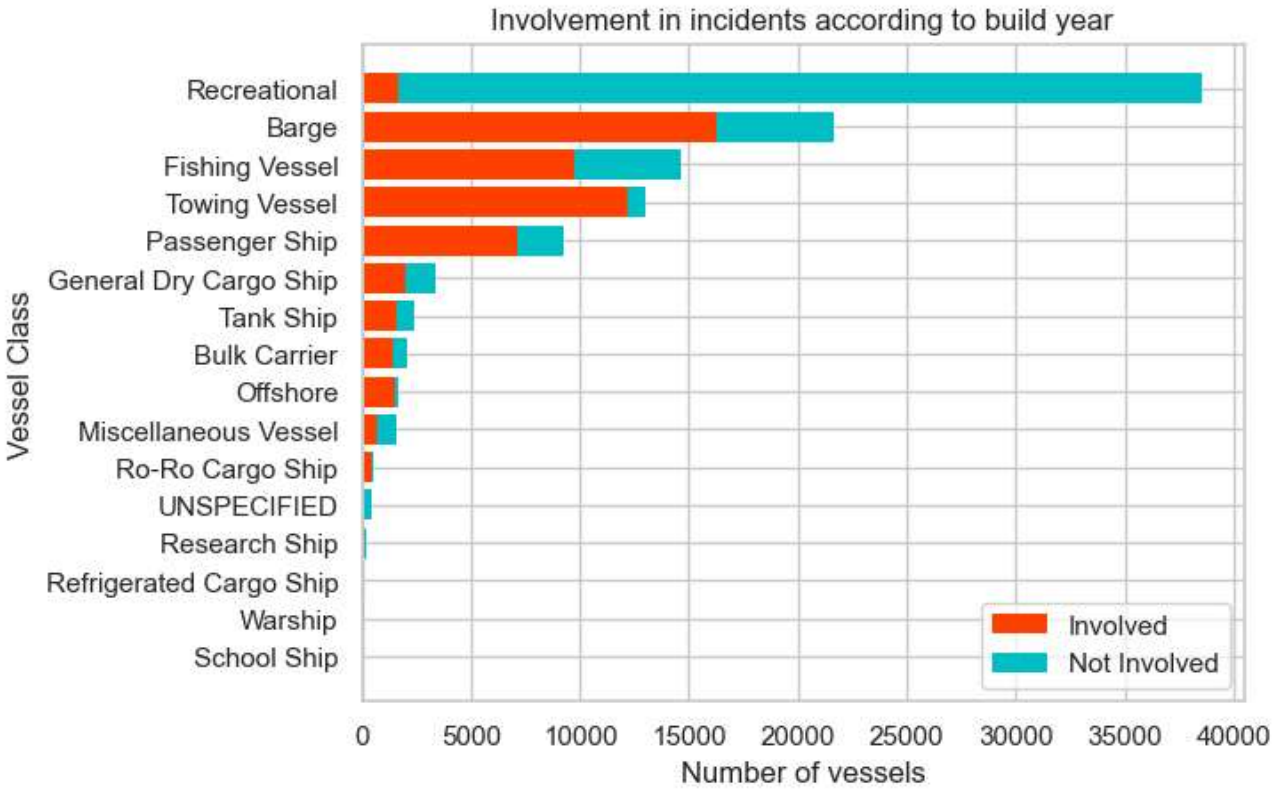
In [19]:
```
# Filter data:
filtered_df = (VesselBalancedSample
               [(VesselBalancedSample['build_year'] >= 1800) & (VesselBalancedSample['bu
               .groupby('build_year')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=False)
               .reset_index())

# Plot bars in stack manner
plt.figure(figsize=(10, 6))
inv_bars = plt.bar(filtered_df['build_year'], filtered_df['involved'],
                   color='orangered', edgecolor='none')
notinv_bars = plt.bar(filtered_df['build_year'], filtered_df['not_involved'],
                      bottom=filtered_df['involved'],
```

```
                      color='#00bfc4', edgecolor='none')

# Customize plot
plt.title('Involvement in incidents according to build year')
plt.xlabel('build year')
plt.ylabel('Number of vessels')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'), loc='upper left'
plt.show()
```



3.2. Incident involvement / vessel_class

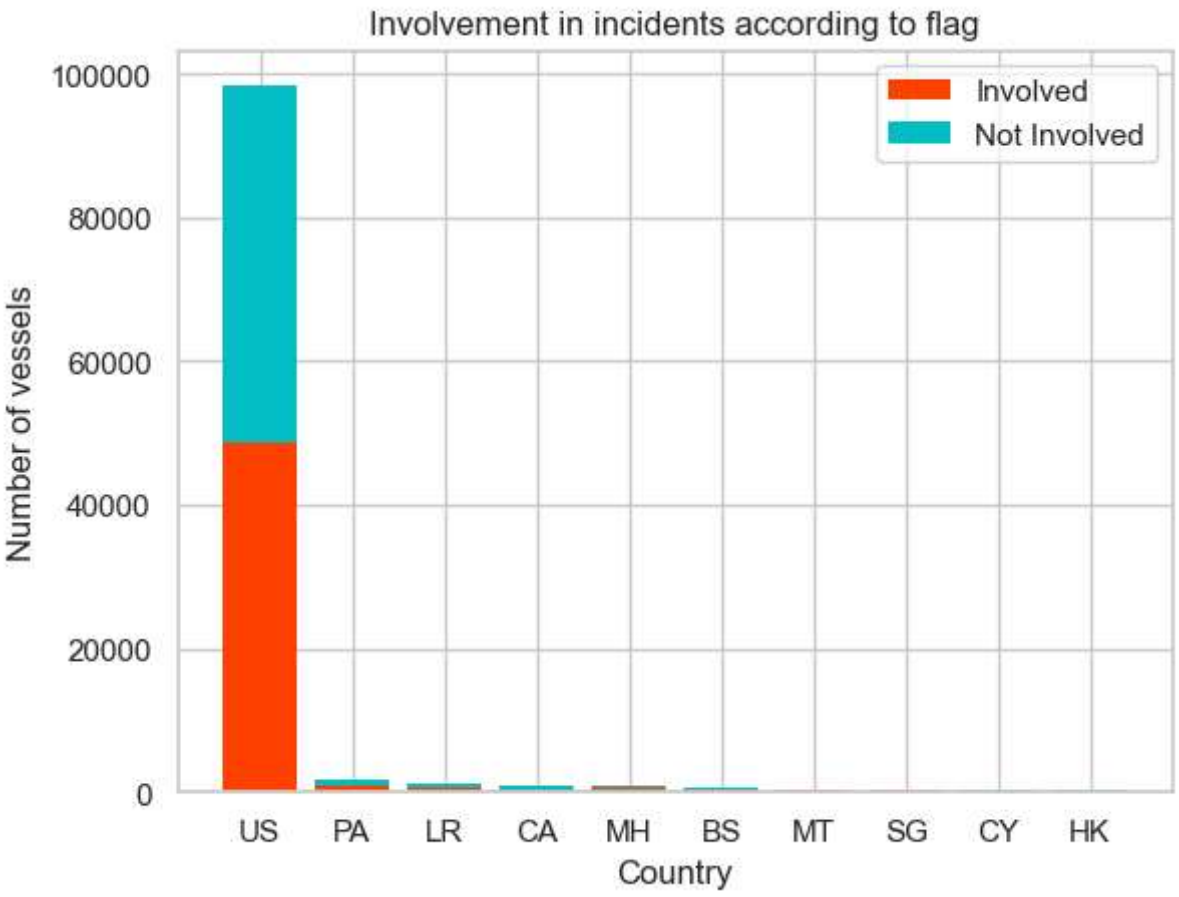
In [20]:
```
# Filter data:
filtered_df = (VesselBalancedSample
               .groupby('vessel_class')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=True)
               .reset_index())

# Plot bars in stack manner
inv_bars = plt.barh(filtered_df['vessel_class'], filtered_df['involved'],
                    color='orangered', edgecolor='none')
notinv_bars = plt.barh(filtered_df['vessel_class'], filtered_df['not_involved'],
                       left=filtered_df['involved'],
                       color='#00bfc4', edgecolor='none')

# Customize plot
plt.title('Involvement in incidents according to build year')
plt.xlabel('Number of vessels')
plt.ylabel('Vessel Class')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'), loc='lower right
plt.show()
```



3.3. Incident involvement / Flag

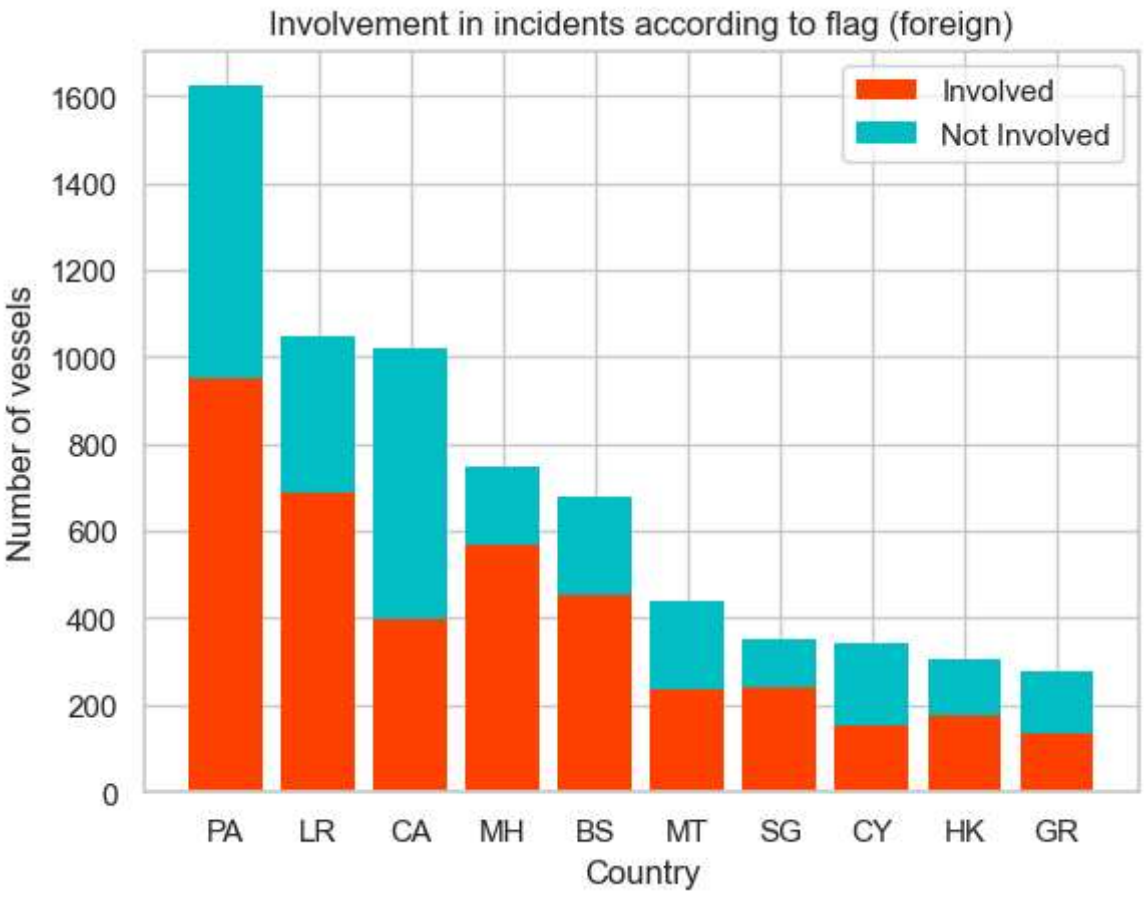
All Flags

In [21]:
```
# Filter data:
filtered_df = (VesselBalancedSample.groupby('flag_abbr')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=False)
               .reset_index()
               .head(10))

# Plot bars in stack manner
inv_bars = plt.bar(filtered_df['flag_abbr'], filtered_df['involved'],
                   color='orangered', edgecolor='none')
notinv_bars = plt.bar(filtered_df['flag_abbr'], filtered_df['not_involved'],
                      bottom=filtered_df['involved'],
                      color='#00bfc4', edgecolor='none')

# Customize plot
plt.title('Involvement in incidents according to flag')
plt.xlabel('Country')
plt.ylabel('Number of vessels')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'))
plt.show()
```



Foreign Flags

In [22]:
```
# Filter data:
filtered_df = (VesselBalancedSample
               [VesselBalancedSample['flag_abbr'] != "US"]
               .groupby('flag_abbr')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=False)
               .reset_index()
               .head(10))

# Plot bars in stack manner
inv_bars = plt.bar(filtered_df['flag_abbr'], filtered_df['involved'],
                   color='orangered', edgecolor='none')
notinv_bars = plt.bar(filtered_df['flag_abbr'], filtered_df['not_involved'],
                      bottom=filtered_df['involved'],
                      color='#00bfc4', edgecolor='none')

# Customize plot
plt.title('Involvement in incidents according to flag (foreign)')
plt.xlabel('Country')
plt.ylabel('Number of vessels')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'))
plt.show()
```



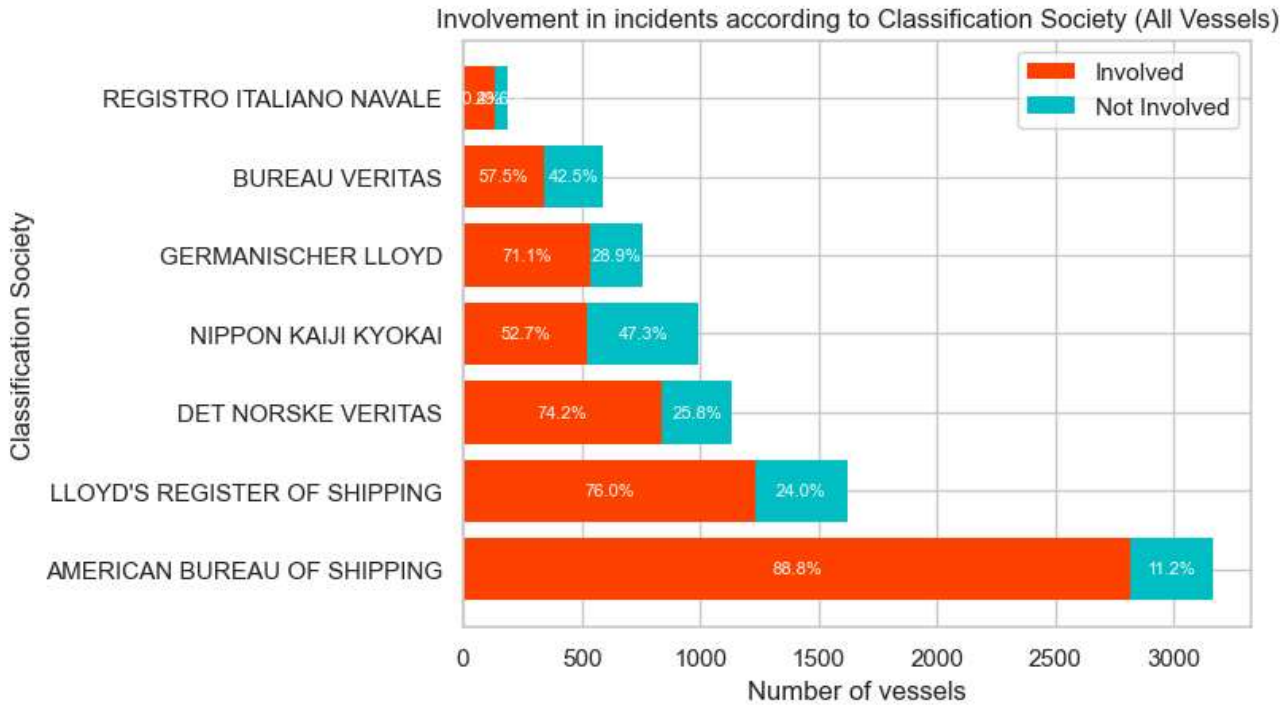
3.4. Incident involvement / classification_society

All vessels

In [23]:

```
# Filter data:
filtered_df = (VesselBalancedSample
               [VesselBalancedSample['classification_society'] != "UNSPECIFIED"]
               .groupby('classification_society')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=False)
               .reset_index()
               .head(7))

# Plot bars in stack manner
fig, ax = plt.subplots()
inv_bars = ax.barh(filtered_df['classification_society'], filtered_df['involved'],
                   color='orangered', edgecolor='none')
notinv_bars = ax.barh(filtered_df['classification_society'], filtered_df['not_involved']
                      left=filtered_df['involved'],
                      color='#00bfc4', edgecolor='none')

# Percentages
for i, bar in enumerate(inv_bars):
    percentage = '{:.1f}%'.format((filtered_df['involved'][i] / filtered_df['total'][i])
    ax.text(0 + bar.get_width() / 2, bar.get_y() + bar.get_height() / 2, percentage,
            va='center', ha='center', color='white', size='8')

for i, bar in enumerate(notinv_bars):
    percentage = '{:.1f}%'.format((filtered_df['not_involved'][i] / filtered_df['total']
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_y() + bar.get_height() / 2, perce
            va='center', ha='center', color='white', size='8')

# Customize plot
```

```
plt.title('Involvement in incidents according to Classification Society (All Vessels)')
plt.xlabel('Number of vessels')
plt.ylabel('Classification Society')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'))
plt.show()
```

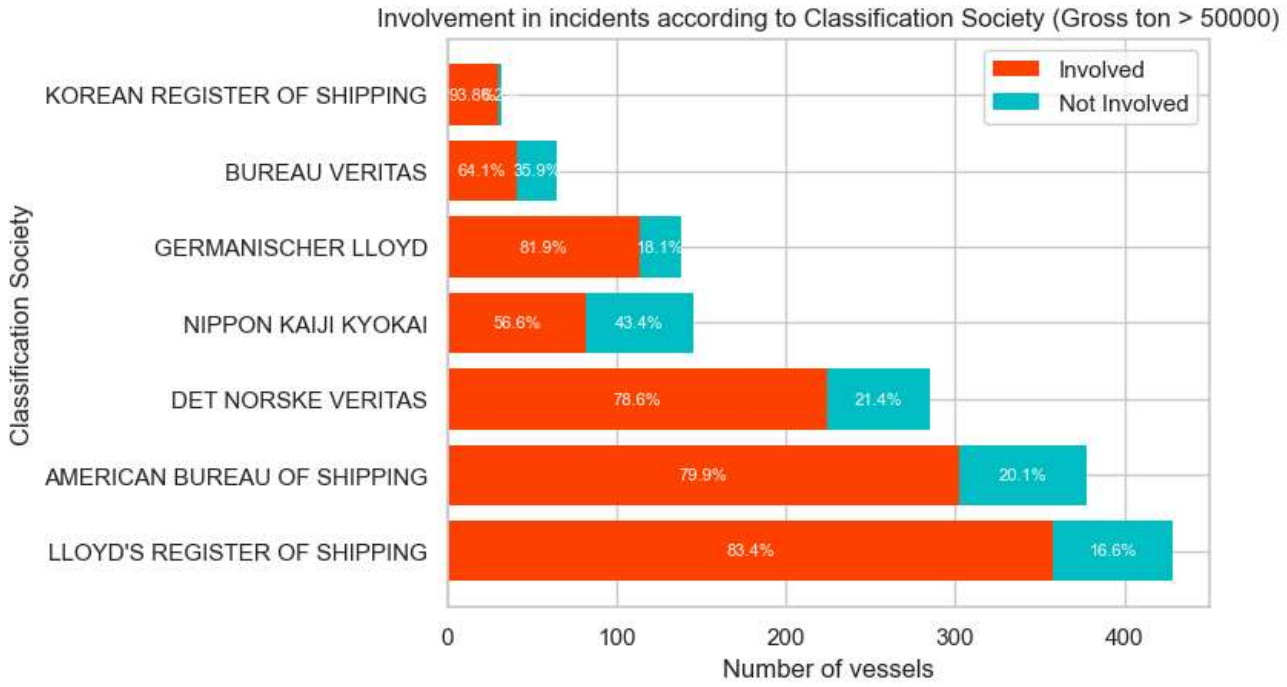


gross_ton > 50000 GPT

In [24]:

```
# Filter data:
filtered_df = (VesselBalancedSample
               [(VesselBalancedSample['classification_society'] != "UNSPECIFIED") &
                (VesselBalancedSample['gross_ton'] >= 50000)]
                .groupby('classification_society')['involved']
                .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='coun
                .sort_values(by='total', ascending=False)
                .reset_index()
                .head(7))

# Plot bars in stack manner
fig, ax = plt.subplots()
inv_bars = ax.barh(filtered_df['classification_society'], filtered_df['involved'],
                   color='orangered', edgecolor='none')
notinv_bars = ax.barh(filtered_df['classification_society'], filtered_df['not_involved']
                      left=filtered_df['involved'],
                      color='#00bfc4', edgecolor='none')

# Percentages
for i, bar in enumerate(inv_bars):
    percentage = '{:.1f}%'.format((filtered_df['involved'][i] / filtered_df['total'][i])
    ax.text(0 + bar.get_width() / 2, bar.get_y() + bar.get_height() / 2, percentage,
            va='center', ha='center', color='white', size='8')

for i, bar in enumerate(notinv_bars):
    percentage = '{:.1f}%'.format((filtered_df['not_involved'][i] / filtered_df['total']
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_y() + bar.get_height() / 2, perce
            va='center', ha='center', color='white', size='8')

# Customize plot
plt.title('Involvement in incidents according to Classification Society (Gross ton > 500
plt.xlabel('Number of vessels')
```

```
plt.ylabel('Classification Society')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'))
plt.show()
```

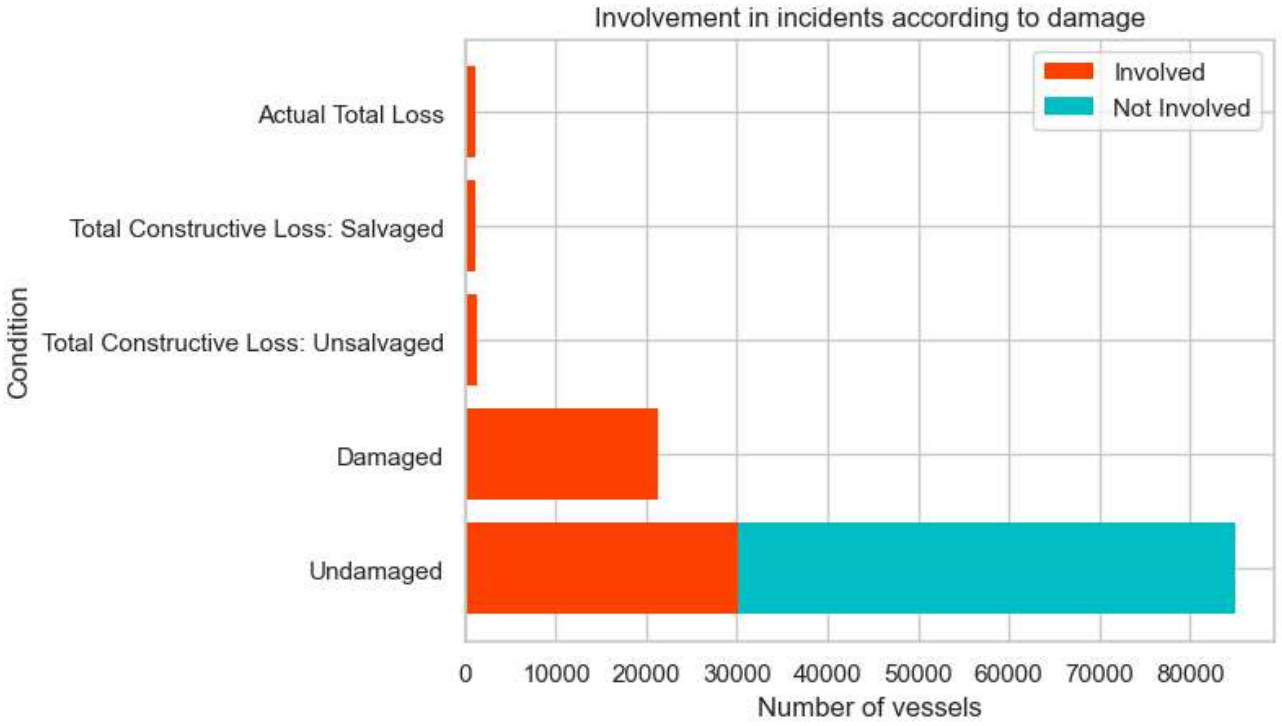


3.5. Incident involvement / SOLAS Membership

In [25]:
```
# Filter data:
filtered_df = (VesselBalancedSample
               .groupby('solas_desc')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=False)
               .reset_index())

# Plot bars in stack manner
inv_bars = plt.bar(filtered_df['solas_desc'], filtered_df['involved'],
                   color='orangered', edgecolor='none')
notinv_bars = plt.bar(filtered_df['solas_desc'], filtered_df['not_involved'],
                      bottom=filtered_df['involved'],
                      color='#00bfc4', edgecolor='none')

# Customize plot
plt.title('Involvement in incidents according to SOLAS membership')
plt.xlabel(None)
plt.ylabel('Number of vessels')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'))
plt.show()
```

3.6. Incident involvement / damage_status

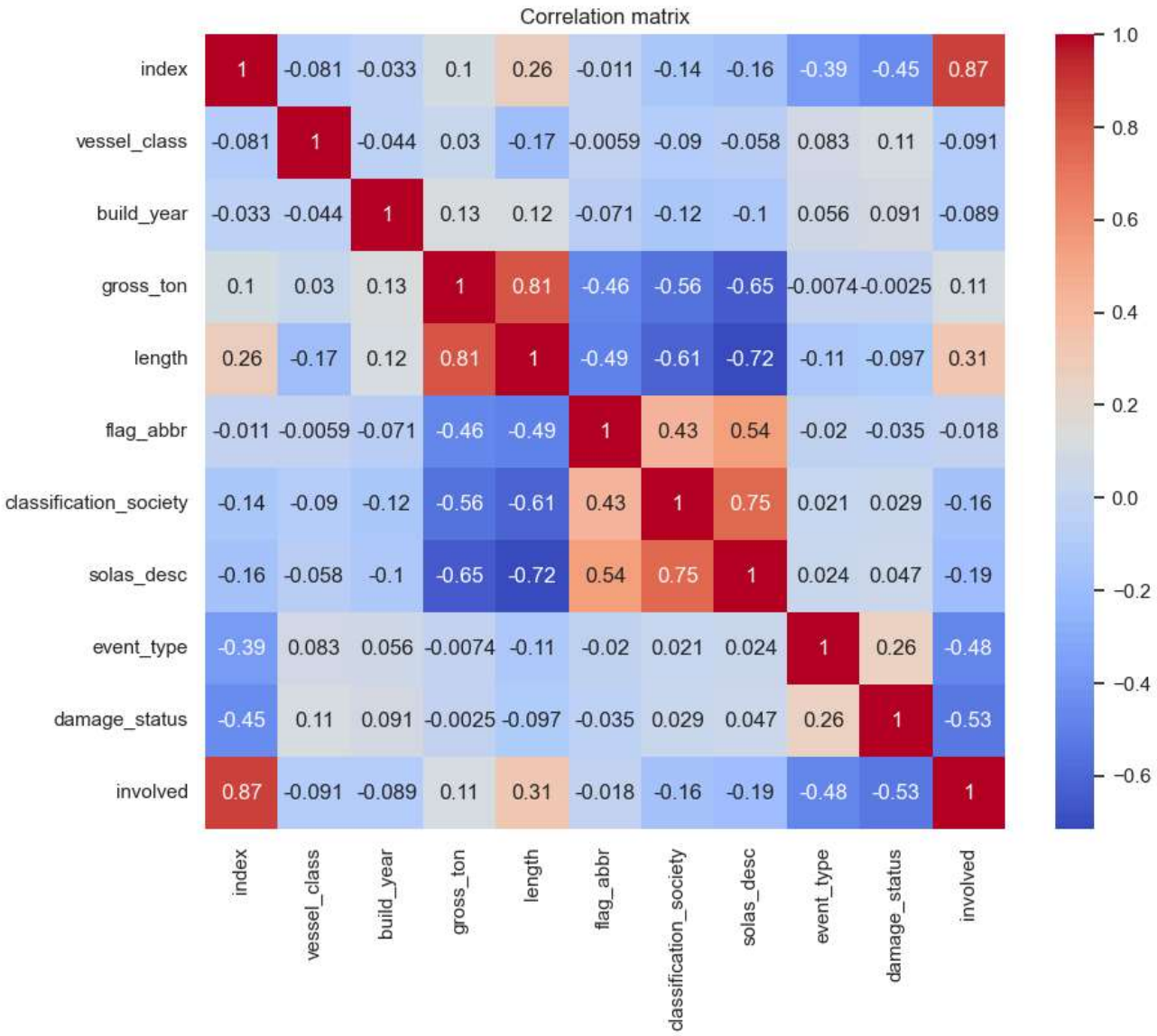
In [26]:
```
# Filter data:
filtered_df = (VesselBalancedSample
               .groupby('damage_status')['involved']
               .agg(involved='sum', not_involved=lambda x: len(x) - sum(x), total='count
               .sort_values(by='total', ascending=False)
               .reset_index())

# Plot bars in stack manner
inv_bars = plt.barh(filtered_df['damage_status'], filtered_df['involved'],
                    color='orangered', edgecolor='none')
notinv_bars = plt.barh(filtered_df['damage_status'], filtered_df['not_involved'],
                       left=filtered_df['involved'],
                       color='#00bfc4', edgecolor='none')

# Customize plot
plt.title('Involvement in incidents according to damage')
plt.xlabel('Number of vessels')
plt.ylabel('Condition')
plt.legend((inv_bars[0], notinv_bars[0]), ('Involved', 'Not Involved'))
plt.show()
```



4. Correlations

4.1. Correlation Matrix

In [27]:
```
# Drop unvaluable variables
VesselBalancedSample = VesselBalancedSample.drop(['vessel_id', 'imo_number', 'vessel_nam

# Convert to numerical using cat.codes
VesselBalancedSample['vessel_class'] = VesselBalancedSample['vessel_class'].astype('cate
VesselBalancedSample['flag_abbr'] = VesselBalancedSample['flag_abbr'].astype('category')
VesselBalancedSample['classification_society'] = VesselBalancedSample['classification_so
VesselBalancedSample['solas_desc'] = VesselBalancedSample['solas_desc'].astype('category
VesselBalancedSample['event_type'] = VesselBalancedSample['event_type'].astype('category
VesselBalancedSample['damage_status'] = VesselBalancedSample['damage_status'].astype('ca

# Convert all to numeric
VesselBalancedSample = VesselBalancedSample.apply(lambda x: pd.to_numeric(x, errors='coe

# Heatmap for correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(VesselBalancedSample.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation matrix')
plt.show()
```



5. Extra: Automatic EDA report

In [28]:
```
# Create ydata_profiling report
profile = ProfileReport(VesselBalancedSample, title='VesselBalancedSample: EDA')

# Export inform
if file_export_enabled :
    profile.to_file("Exported Reports/VesselBalancedSample_EDA.html")
else:
    print('EDA report already exported')
```

```
EDA report already exported
```