In [51]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import seaborn as sns
from sklearn import metrics
%matplotlib inline
```

In [11]:
```
print("Titanic survivial prediction Using Logistic Regression Model from scikit-
print("☕ Grab a cup of coffee and watch the magic happen!")

# Load the dataset
train = pd.read_csv('../data/titanic_train.csv')
train.head
```

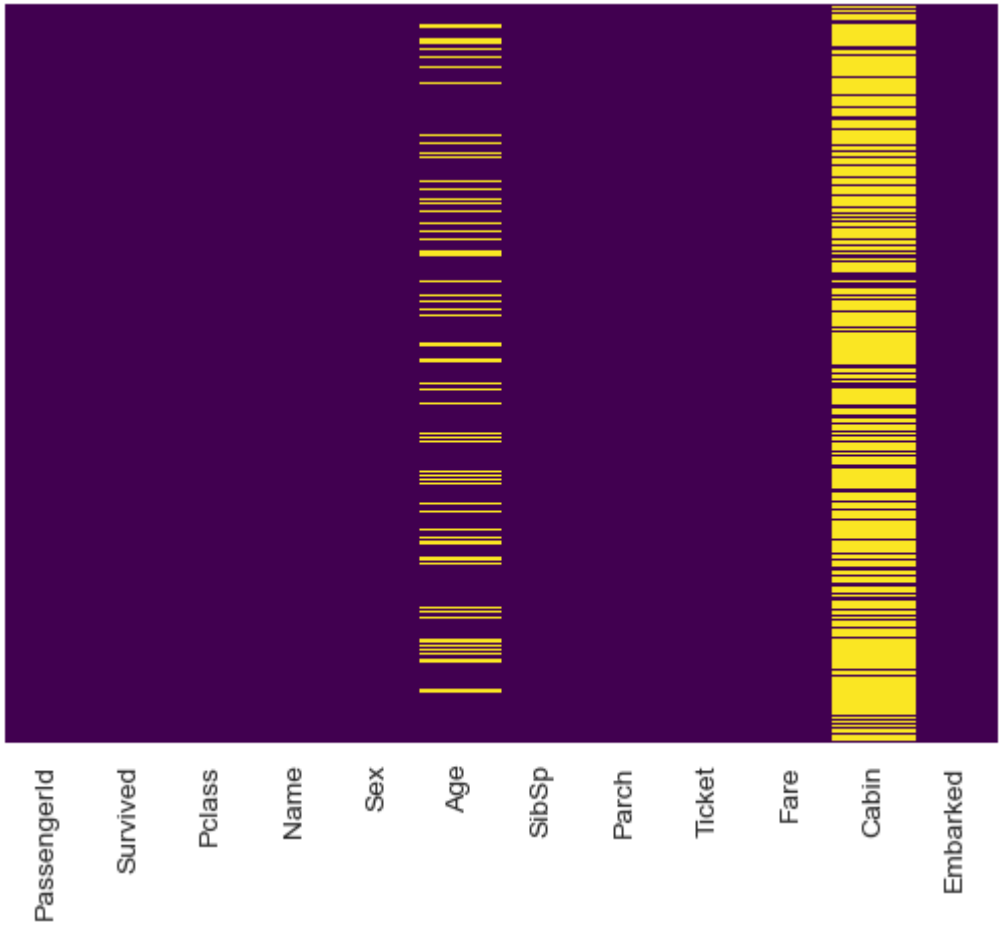
```
Titanic survivial prediction Using Logistic Regression Model from scikit-learn...
☕ Grab a cup of coffee and watch the magic happen!
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0

In [66]:
```
#To check if any datapoints are null ; the particular chart below shows age data
sns.heatmap(train.isnull(), yticklabels=False, cbar=False, cmap='viridis')
```

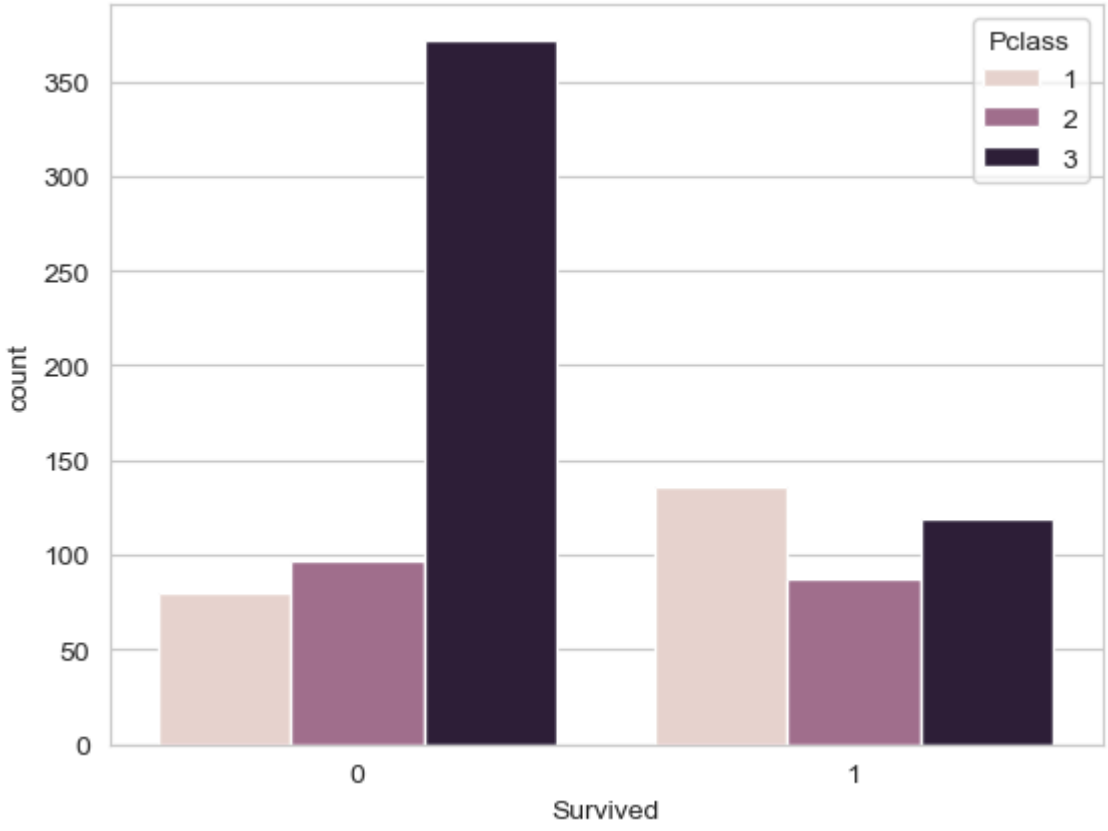
Out[66]: <Axes: >



In [19]:
```
sns.set_style('whitegrid')
```

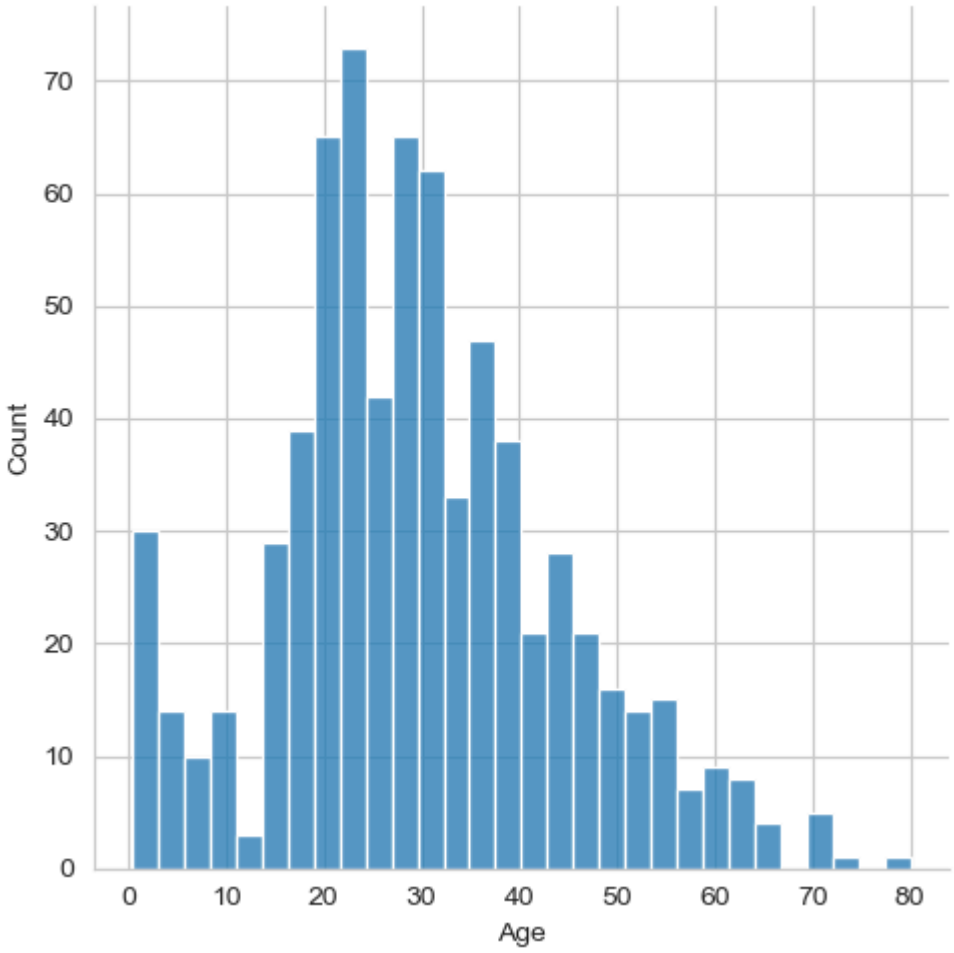
In [70]:
```
sns.countplot(x='Survived',hue='Pclass', data=train)
```

Out[70]:
```
<Axes: xlabel='Survived', ylabel='count'>
```



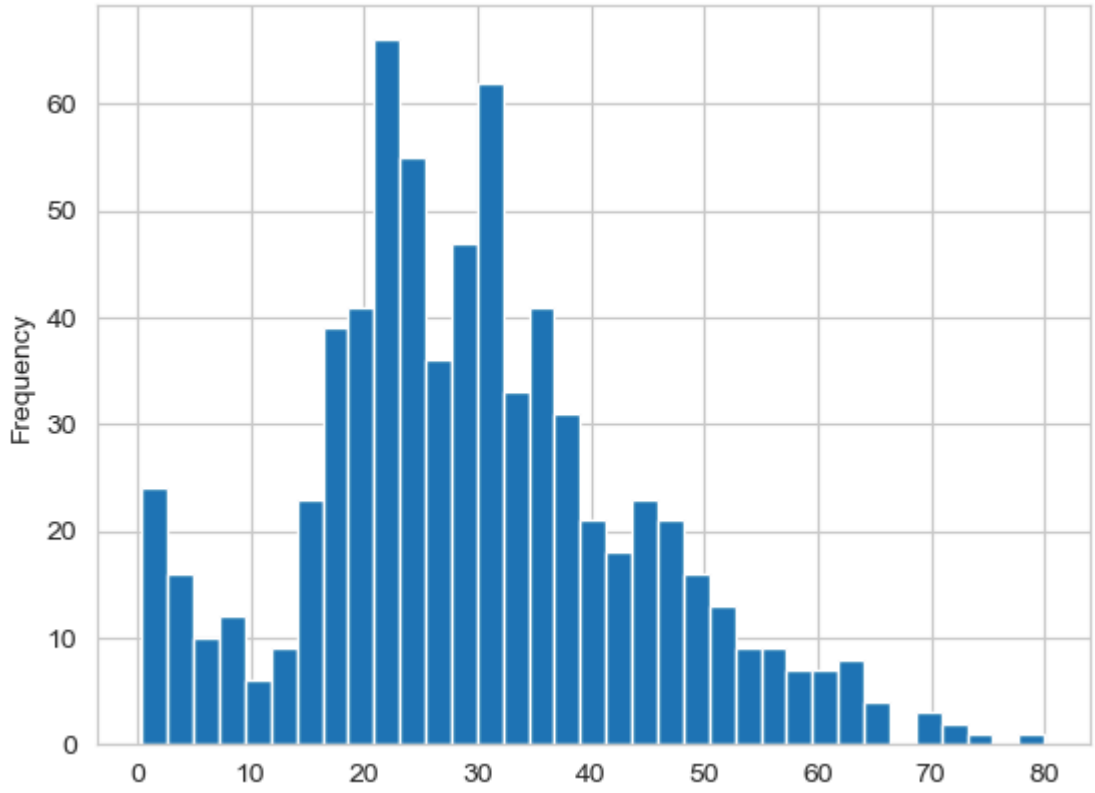
In [74]:
```
sns.displot(train['Age'].dropna(), kde=False, bins=30)
```

Out[74]:
```
<seaborn.axisgrid.FacetGrid at 0x217308d7bf0>
```



In [35]:
```
train['Age'].plot.hist(bins=35)
```

Out[35]: <Axes: ylabel='Frequency'>

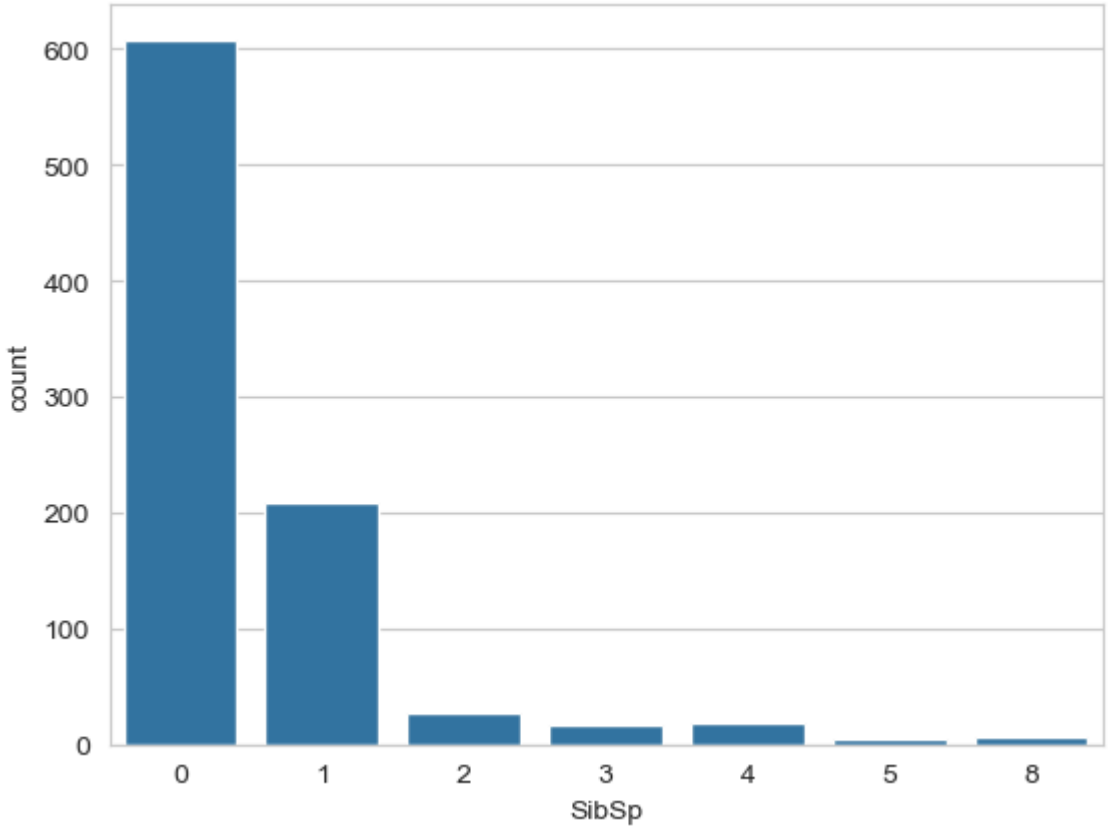


In [37]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

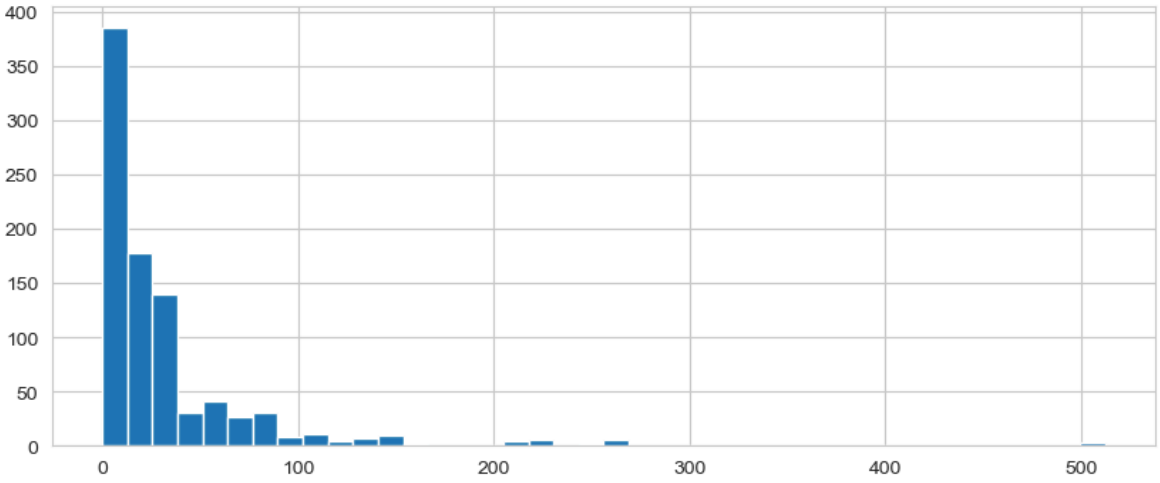
In [39]:
```
sns.countplot(x='SibSp', data=train)
```

Out[39]: <Axes: xlabel='SibSp', ylabel='count'>



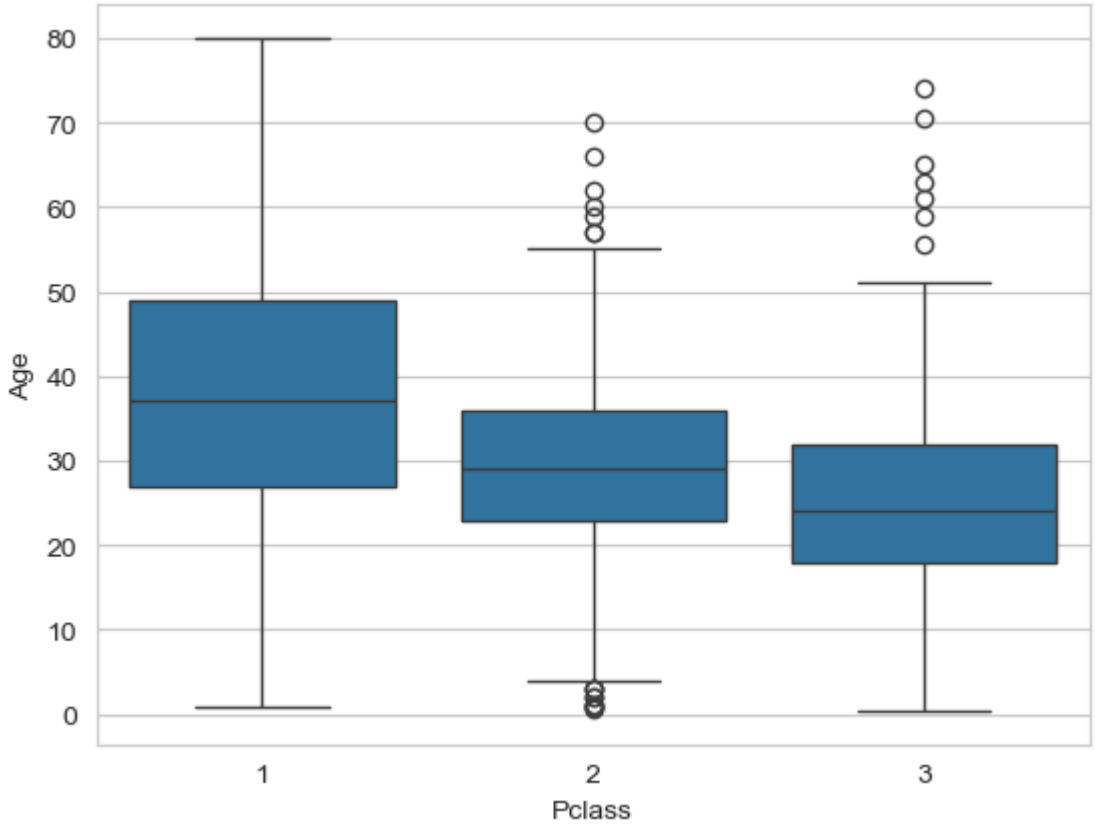
In [43]:
```
train['Fare'].hist(bins=40, figsize=(10,4))
```

Out[43]: <Axes: >



In [78]:
```
sns.boxplot(x='Pclass', y='Age', data=train)
```

Out[78]: <Axes: xlabel='Pclass', ylabel='Age'>

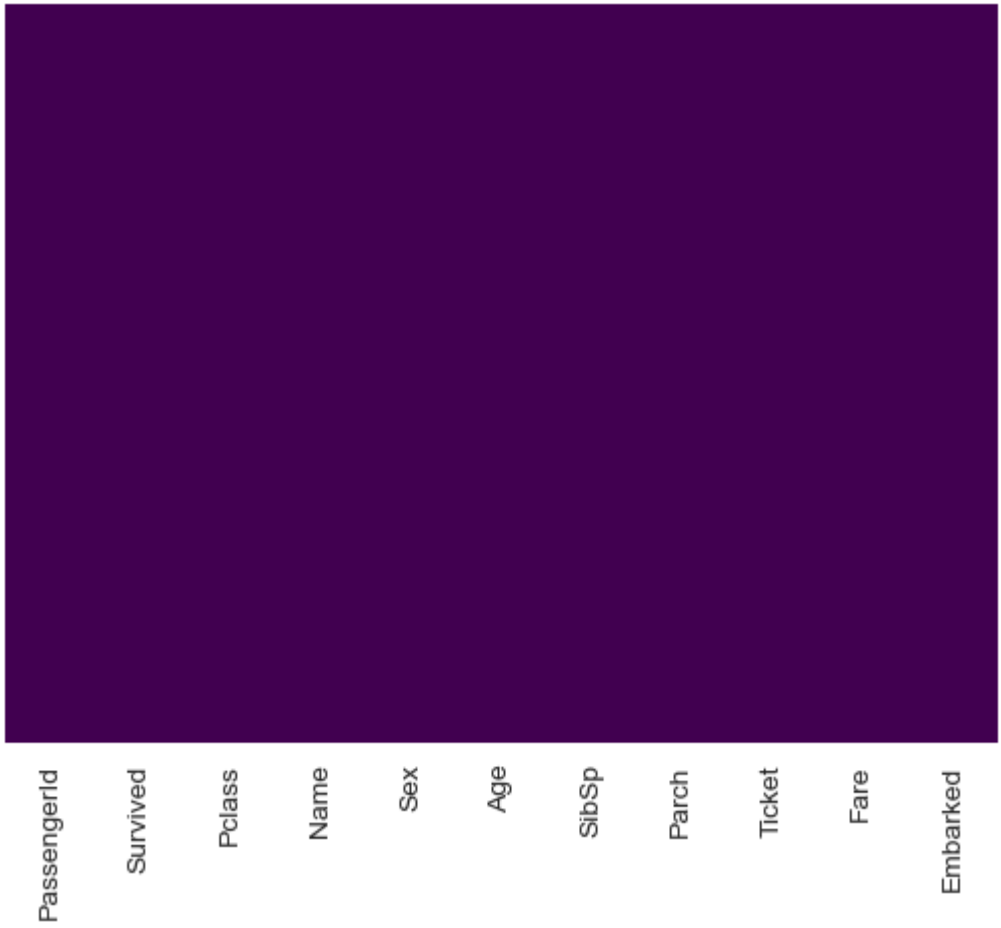


In [98]:
```
def impute_age(cols):
    Age = cols.iloc[0]
    Pclass = cols.iloc[1]
    if pd.isnull(Age):  # Check if Age is NaN
        if Pclass == 1:
            return 37  # Impute age for Pclass 1
        elif Pclass == 2:
            return 29  # Impute age for Pclass 2
        else:
            return 24  # Impute age for Pclass 3 or any other condition
    else:
        return Age  # Return the original Age if it's not NaN
```

In [102…
```
train['Age'] = train[['Age', 'Pclass']].apply(impute_age,axis=1)
```

In [104…
```
sns.heatmap(train.isnull(), yticklabels=False, cbar=False, cmap = 'viridis')
```

Out[104… <Axes: >



In [94]:
```
train.drop('Cabin', axis=1, inplace=True)
```

In [132…
```
sex = pd.get_dummies(train['Sex'], drop_first=True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File C:\learnings\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'Sex'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[132], line 1
----> 1 sex = pd.get_dummies(train['Sex'], drop_first=True)

File C:\learnings\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__getitem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File C:\learnings\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'Sex'
```

In [118…
```
embark = pd.get_dummies(train['Embarked'], drop_first=True)
```

In [120…
```
train = pd.concat([train,sex, embark], axis=1)
```

In [122…
```
train.head()
```

Out[122…

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	F
0	1	0	3	Braund, Mr. Owen Harris	male	24.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	37.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	24.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	37.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	24.0	0	0	373450	8.0

In [134…

```
train.drop(['Sex', 'Embarked', 'Name', 'Ticket'], axis=1, inplace=True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[134], line 1
----> 1 train.drop(['Sex', 'Embarked', 'Name', 'Ticket'], axis=1, inplace=True)

File C:\learnings\Lib\site-packages\pandas\core\frame.py:5581, in DataFrame.drop
(self, labels, axis, index, columns, level, inplace, errors)
   5433 def drop(
   5434     self,
   5435     labels: IndexLabel | None = None,
   (...)
   5442     errors: IgnoreRaise = "raise",
   5443 ) -> DataFrame | None:
   5444     """
   5445     Drop specified labels from rows or columns.
   5446
   (...)
   5579             weight  1.0     0.8
   5580     """
-> 5581     return super().drop(
   5582         labels=labels,
   5583         axis=axis,
   5584         index=index,
   5585         columns=columns,
   5586         level=level,
   5587         inplace=inplace,
   5588         errors=errors,
   5589     )

File C:\learnings\Lib\site-packages\pandas\core\generic.py:4788, in NDFrame.drop
(self, labels, axis, index, columns, level, inplace, errors)
   4786 for axis, labels in axes.items():
   4787     if labels is not None:
-> 4788         obj = obj._drop_axis(labels, axis, level=level, errors=errors)
   4790 if inplace:
   4791     self._update_inplace(obj)

File C:\learnings\Lib\site-packages\pandas\core\generic.py:4830, in NDFrame._drop
_axis(self, labels, axis, level, errors, only_slice)
   4828         new_axis = axis.drop(labels, level=level, errors=errors)
   4829     else:
-> 4830         new_axis = axis.drop(labels, errors=errors)
   4831     indexer = axis.get_indexer(new_axis)
   4833 # Case for non-unique axis
   4834 else:

File C:\learnings\Lib\site-packages\pandas\core\indexes\base.py:7070, in Index.dr
op(self, labels, errors)
   7068 if mask.any():
   7069     if errors != "ignore":
-> 7070         raise KeyError(f"{labels[mask].tolist()} not found in axis")
   7071     indexer = indexer[~mask]
   7072 return self.delete(indexer)

KeyError: "['Sex', 'Embarked', 'Name', 'Ticket'] not found in axis"
```

In [136…

```
train[['male', 'Q', 'S']] = train[['male', 'Q', 'S']].astype(int)
train.head()
```

Out[136…

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	0	3	24.0	1	0	7.2500	1	0	1
1	2	1	1	37.0	1	0	71.2833	0	0	0
2	3	1	3	24.0	0	0	7.9250	0	0	1
3	4	1	1	37.0	1	0	53.1000	0	0	1
4	5	0	3	24.0	0	0	8.0500	1	0	1

In [138…
```
train.drop('PassengerId', axis=1, inplace=True)
```

In [140…
```
train.head()
```

Out[140…

	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	0	3	24.0	1	0	7.2500	1	0	1
1	1	1	37.0	1	0	71.2833	0	0	0
2	1	3	24.0	0	0	7.9250	0	0	1
3	1	1	37.0	1	0	53.1000	0	0	1
4	0	3	24.0	0	0	8.0500	1	0	1

In [142…
```
x = train.drop('Survived', axis=1)
y = train['Survived']
```

In [144…
```
from sklearn.model_selection import train_test_split
```

In [146…
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random
```

In [148…
```
lgrm = LogisticRegression()
lgrm.fit(x_train, y_train)
```

```
C:\learnings\Lib\site-packages\sklearn\linear_model\_logistic.py:469: Convergence
Warning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[148…

▾ LogisticRegression

LogisticRegression()

In [150…
```
predictions = lgrm.predict(x_test)
```

In [152…
```
from sklearn.metrics import classification_report
```

In [154…
```
print(classification_report(y_test, predictions))
```

```
                   precision    recall  f1-score   support

               0       0.78      0.88      0.82       169
               1       0.80      0.67      0.73       126

        accuracy                           0.79       295
       macro avg       0.79      0.77      0.78       295
    weighted avg       0.79      0.79      0.78       295
```

In [156…
```
from sklearn.metrics import confusion_matrix
```

In [162…
```
cf = confusion_matrix(y_test, predictions)
```

In [164…
```
cm_df = pd.DataFrame(cf,
                     index=['Actual Negative', 'Actual Positive'],
                     columns=['Predicted Negative', 'Predicted Positive'])
```

In [168…
```
cm_df.head()
```

Out[168…

	Predicted Negative	Predicted Positive
Actual Negative	148	21
Actual Positive	42	84

In [ ]:
```
```