```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import plotly.express as px

from sklearn.preprocessing import LabelEncoder
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

from sklearn.metrics import classification_report

import warnings
warnings.filterwarnings('ignore')
```

```
titanic = pd.read_csv('/content/drive/MyDrive/MOUNTT/tested.csv')
```

```
titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Far
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.829
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.000

```
titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Survived     418 non-null    int64
 2   Pclass       418 non-null    int64
 3   Name         418 non-null    object
 4   Sex          418 non-null    object
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64
 7   Parch        418 non-null    int64
 8   Ticket       418 non-null    object
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object
 11  Embarked     418 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

```
titanic.isna().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

```
# Handling the null values

columns = ['Age', 'Fare']
for col in columns:
    titanic[col].fillna(titanic[col].median(), inplace = True)

titanic['Cabin'].fillna('Unknown', inplace=True)
```

```
#checking duplicate values

dup = titanic.duplicated().sum()
print("The number of duplicated values in the dataset are: ", dup)
```

```
The number of duplicated values in the dataset are:  0
```

```
#Checking if there are any typos

for col in titanic.select_dtypes(include = "object"):
    print(f"Name of Column: {col}")
    print(titanic[col].unique())
    print('\n', '-'*60, '\n')
```

```
 '19877' 'STON/O 2. 3101288' '347471' 'A/5. 3338' '11778' '228414'
 '365235' '347070' '2625' 'C 4001' '330920' '383162' '3410' '248734'
 '237734' '330968' 'PC 17531' '329944' '2680' '2681' 'PP 9549' '13050'
 'SC/AH 29037' 'C.A. 33595' '367227' '392095' '368783' '371362' '350045'
 '367226' '211535' '342441' 'STON/OQ. 369943' '113780' '4133' '2621'
 '349226' '350409' '2656' '248659' 'SOTON/OQ 392083' 'CA 2144' '113781'
 '244358' '17475' '345763' '17463' 'SC/A4 23568' '113791' '250651' '11767'
 '349255' '3701' '350405' '347077' 'S.O./P.P. 752' '347469' '110489'
 'SOTON/O.Q. 3101315' '335432' '2650' '220844' '343271' '237393' '315153'
 'PC 17591' 'W./C. 6608' '17770' '7548' 'S.O./P.P. 251' '2670' '2673'
 '29750' 'C.A. 33112' '230136' 'PC 17756' '233478' '113773' '7935'
 'PC 17558' '239059' 'S.O./P.P. 2' 'A/4 48873' 'CA. 2343' '28221' '226875'
 '111163' 'A/5. 851' '235509' '28220' '347465' '16966' '347066'
 'C.A. 31030' '65305' '36568' '347080' 'PC 17757' '26360' 'C.A. 34050'
 'F.C. 12998' '9232' '28034' 'PC 17613' '349250' 'SOTON/O.Q. 3101308'
 'S.O.C. 14879' '347091' '113038' '330924' '36928' '32302' 'SC/PARIS 2148'
 '342684' 'W./C. 14266' '350053' 'PC 17606' '2661' '350054' '370368'
 'C.A. 6212' '242963' '220845' '113795' '3101266' '330971' 'PC 17599'
 '350416' '110813' '2679' '250650' 'PC 17761' '112377' '237789' '3470'
 '17464' '26707' 'C.A. 34651' 'SOTON/O2 3101284' '13508' '7266' '345775'
 'C.A. 42795' 'AQ/4 3130' '363611' '28404' '345501' '345572' '350410'
 'C.A. 34644' '349235' '112051' 'C.A. 49867' 'A. 2. 39186' '315095'
 '368573' '370371' '2676' '236853' 'SC 14888' '2926' 'CA 31352'
 'W./C. 14260' '315085' '364859' '370129' 'A/5 21175' 'SOTON/O.Q. 3101314'
 '2655' 'A/5 1478' 'PC 17607' '382650' '2652' '33638' '345771' '349202'
 'SC/Paris 2123' '113801' '347467' '347079' '237735' '315092' '383123'
 '112901' '392091' '12749' '350026' '315091' '2658' 'LP 1588' '368364'
 'PC 17760' 'AQ/3. 30631' 'PC 17569' '28004' '350408' '347075' '2654'
 '244368' '113790' '24160' 'SOTON/O.Q. 3101309' 'PC 17585' '2003' '236854'
 'PC 17580' '2684' '2653' '349229' '110469' '244360' '2675' '2622'
 'C.A. 15185' '350403' 'PC 17755' '348125' '237670' '2688' '248726'
 'F.C.C. 13528' 'PC 17759' 'F.C.C. 13540' '113044' '11769' '1222' '368402'
 '349910' 'S.C./PARIS 2079' '315083' '11765' '2689' '3101295' '112378'
 'SC/PARIS 2147' '28133' '112058' '248746' '315152' '29107' '680' '366713'
 '330910' '364498' '376566' 'SC/PARIS 2159' '349911' '244346' '364858'
 '349909' 'PC 17592' 'C.A. 2673' 'C.A. 30769' '371109' '13567' '347065'
 '21332' '28664' '113059' '17765' 'SC/PARIS 2166' '28666' '334915'
 '365237' '19928' '347086' 'A.5. 3236' 'PC 17758' 'SOTON/O.Q. 3101262'
 '359309' '2668']

 ------------------------------------------------------------

Name of Column: Cabin
['Unknown' 'B45' 'E31' 'B57 B59 B63 B66' 'B36' 'A21' 'C78' 'D34' 'D19'
 'A9' 'D15' 'C31' 'C23 C25 C27' 'F G63' 'B61' 'C53' 'D43' 'C130' 'C132'
 'C101' 'C55 C57' 'B71' 'C46' 'C116' 'F' 'A29' 'G6' 'C6' 'C28' 'C51' 'E46'
 'C54' 'C97' 'D22' 'B10' 'F4' 'E45' 'E52' 'D30' 'B58 B60' 'E34' 'C62 C64'
 'A11' 'B11' 'C80' 'F33' 'C85' 'D37' 'C86' 'D21' 'C89' 'F E46' 'A34' 'D'
 'B26' 'C22 C26' 'B69' 'C32' 'B78' 'F E57' 'F2' 'A18' 'C106' 'B51 B53 B55'
 'D10 D12' 'E60' 'E50' 'E39 E41' 'B52 B54 B56' 'C39' 'B24' 'D28' 'B41'
 'C7' 'D40' 'D38' 'C105']

 ------------------------------------------------------------

Name of Column: Embarked
['Q' 'S' 'C']

 ------------------------------------------------------------
```

```
titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Far
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.829
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.000

```
# Creating a new feature of title from name column based on the pattern found above

titanic['Title'] = titanic['Name'].str.extract(r',\s(.*?)\.')

titanic['Title'] = titanic['Title'].replace('Ms', 'Miss')
titanic['Title'] = titanic['Title'].replace('Dona', 'Mrs')
titanic['Title'] = titanic['Title'].replace(['Col', 'Rev', 'Dr'], 'Rare')
```

```
# Creating another feature of Age group by making bins

bins = [-np.inf, 17, 32, 45, 50, np.inf]
labels = ["Children", "Young", "Mid-Aged", "Senior-Adult", 'Elderly']
titanic['Age_Group'] = pd.cut(titanic['Age'], bins = bins, labels = labels)
```

```
# Generting another new feature of family size

titanic['Family'] = titanic['SibSp'] + titanic['Parch']
```

```
# Dropping non essential coclumns

titanic.drop(['PassengerId', 'Name', 'Ticket'], axis = 1, inplace = True)
```

```
titanic.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked	Title	Age
0	0	3	male	34.5	0	0	7.8292	Unknown	Q	Mr	M
1	1	3	female	47.0	1	0	7.0000	Unknown	S	Mrs	
2	0	2	male	62.0	0	0	9.6875	Unknown	Q	Mr	
3	0	3	male	27.0	0	0	8.6625	Unknown	S	Mr	

```
# Chaning the positon of columns to place them right after their parent column

col_to_move = titanic.pop('Age_Group')
titanic.insert(4, 'Age_Group', col_to_move)

col_to_move = titanic.pop('Family')
titanic.insert(7, 'Family', col_to_move)

titanic['Age_Group'] = titanic['Age_Group'].astype('object')
```

```
titanic.describe()
```

	Survived	Pclass	Age	SibSp	Parch	Family	Far
count	418.000000	418.000000	418.000000	418.000000	418.000000	418.000000	418.00000
mean	0.363636	2.265550	29.599282	0.447368	0.392344	0.839713	35.57653
std	0.481622	0.841838	12.703770	0.896760	0.981429	1.519072	55.85010
min	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000	0.00000
25%	0.000000	1.000000	23.000000	0.000000	0.000000	0.000000	7.89580
50%	0.000000	3.000000	27.000000	0.000000	0.000000	0.000000	14.45420
75%	1.000000	3.000000	35.750000	1.000000	0.000000	1.000000	31.47187
max	1.000000	3.000000	76.000000	8.000000	9.000000	10.000000	512.32920

```
titanic.describe(include = 'O')
```

	Sex	Age_Group	Cabin	Embarked	Title
count	418	418	418	418	418
unique	2	5	77	3	5
top	male	Young	Unknown	S	Mr
freq	266	257	327	270	240

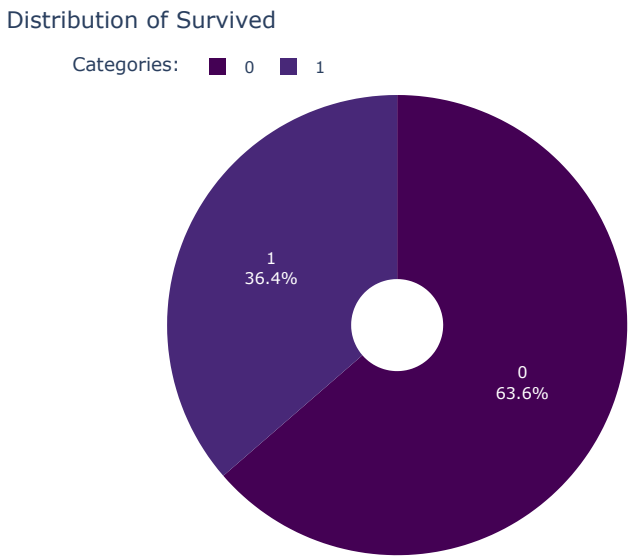
```
titanic.groupby('Sex')[['Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Family', 'Fare']].mean()
```

Sex	Survived	Pclass	Age	SibSp	Parch	Family	Fare
female	1.0	2.144737	29.734145	0.565789	0.598684	1.164474	49.747699
male	0.0	2.334586	29.522218	0.379699	0.274436	0.654135	27.478728

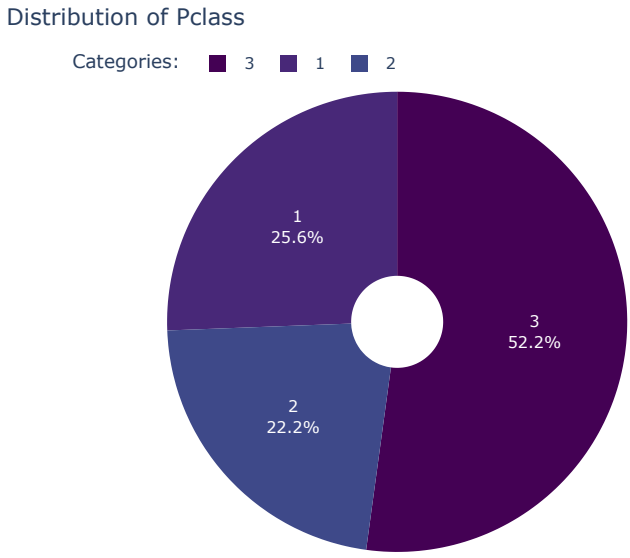
```
titanic.groupby('Embarked')[['Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Family', 'Fare']].mean()
```

Embarked	Survived	Pclass	Age	SibSp	Parch	Family	Fare
C	0.392157	1.794118	33.220588	0.421569	0.382353	0.803922	66.259765
Q	0.521739	2.869565	28.108696	0.195652	0.021739	0.217391	10.957700
S	0.325926	2.340741	28.485185	0.500000	0.459259	0.959259	28.179413

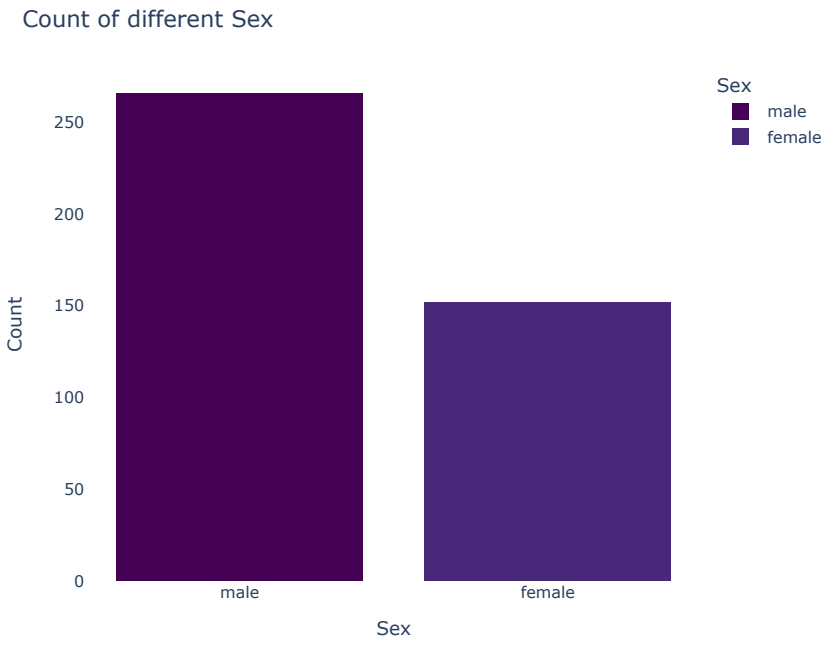
```
survived_counts = titanic['Survived'].value_counts()
fig_surv_perc = px.pie(titanic, names= survived_counts.index,  values = survived_counts.values, title=f'Distribution of Survived', hole=
fig_surv_perc.update_traces(textinfo='percent+label')
fig_surv_perc.update_layout(legend_title_text='Categories:', legend=dict(orientation="h", yanchor="bottom", y=1.02))
fig_surv_perc.show()
```

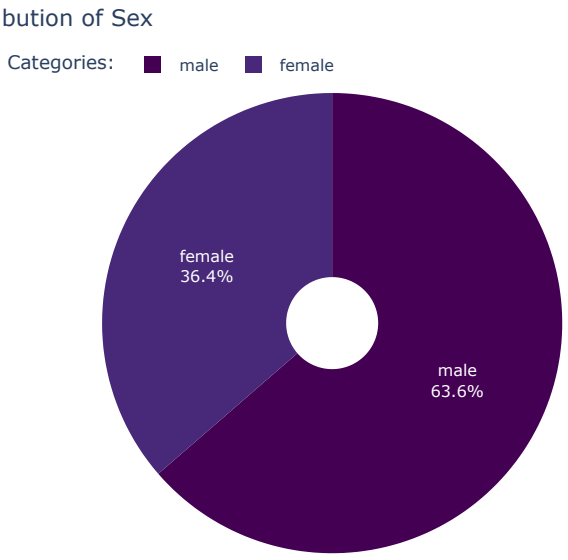



```
pclass_counts = titanic.Pclass.value_counts()
fig_pclass_perc = px.pie(titanic, names= pclass_counts.index, values = pclass_counts.values, title=f'Distribution of Pclass', hole=0.2,
fig_pclass_perc.update_traces(textinfo='percent+label')
fig_pclass_perc.update_layout(legend_title_text='Categories:', legend=dict(orientation="h", yanchor="bottom", y=1.02))
fig_pclass_perc.show()
```

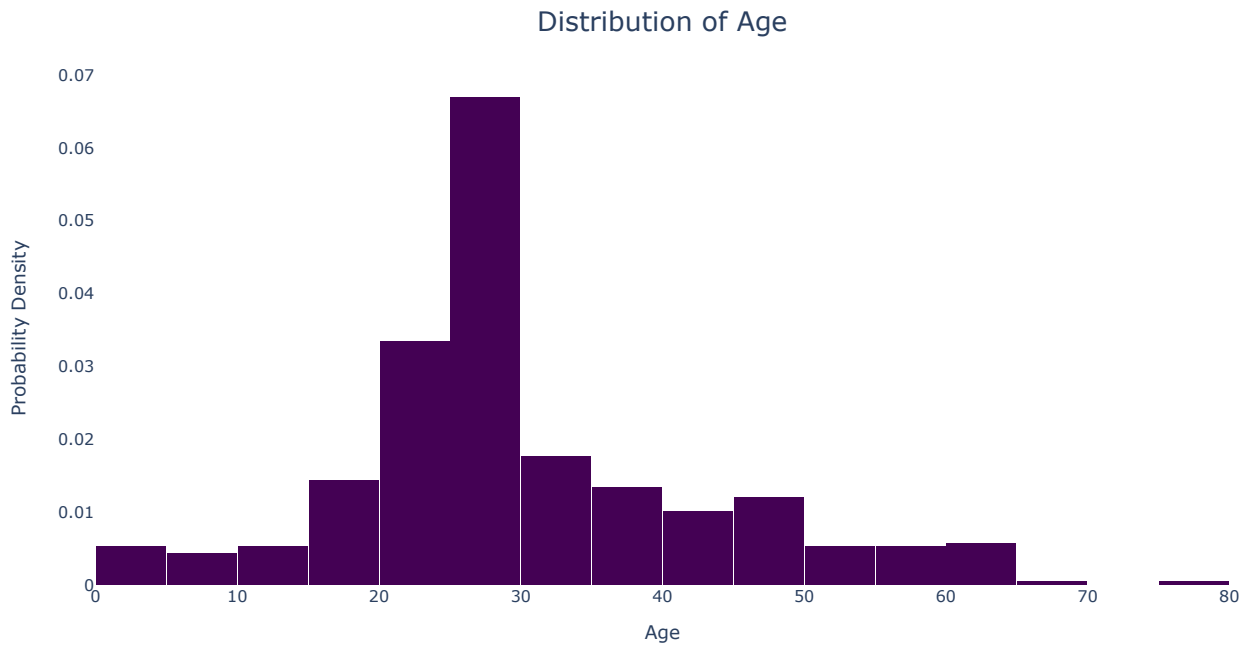



```
fig_sex_count = px.histogram(titanic, x = 'Sex', color = 'Sex', color_discrete_sequence=px.colors.sequential.Viridis)
fig_sex_count.update_layout(title_text='Count of different Sex', xaxis_title='Sex', yaxis_title='Count', plot_bgcolor = 'white')
fig_sex_count.show()

fig_sex_perc = px.pie(titanic, names= 'Sex', title=f'Distribution of Sex', hole=0.2, color_discrete_sequence=px.colors.sequential.Viridi
fig_sex_perc.update_traces(textinfo='percent+label')
fig_sex_perc.update_layout(legend_title_text='Categories:', legend=dict(orientation="h", yanchor="bottom", y=1.02))
fig_sex_perc.show()
```

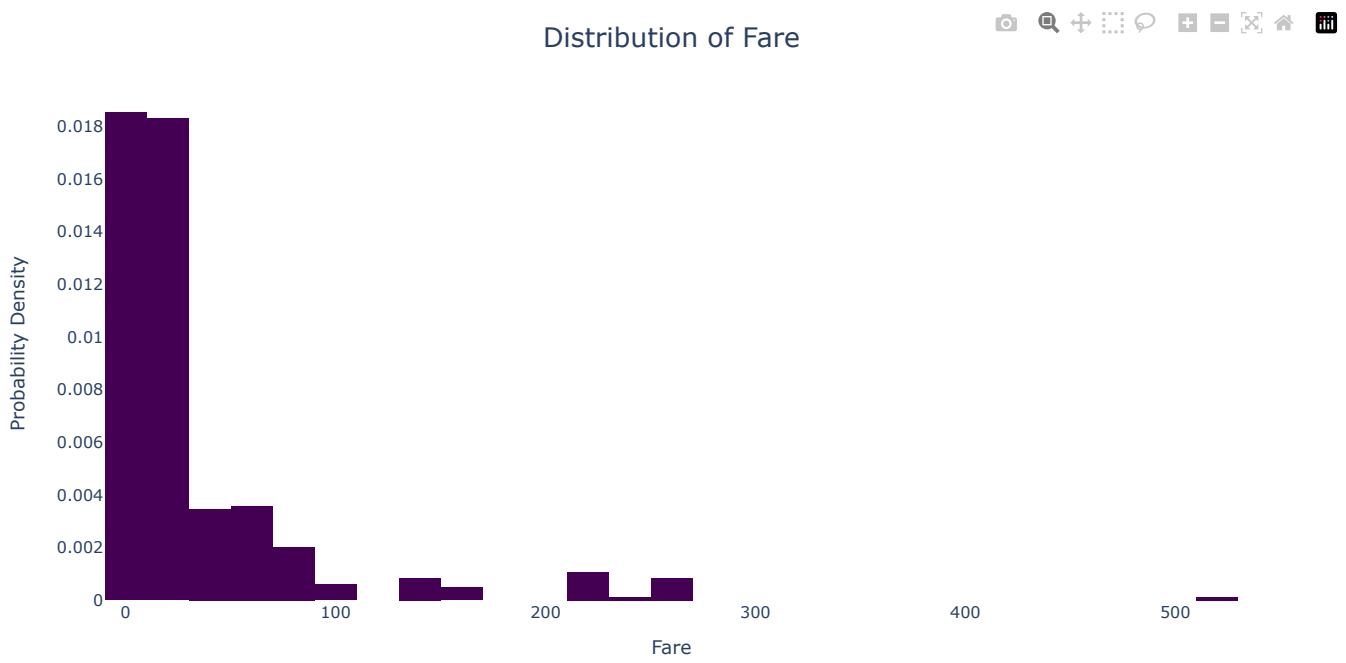




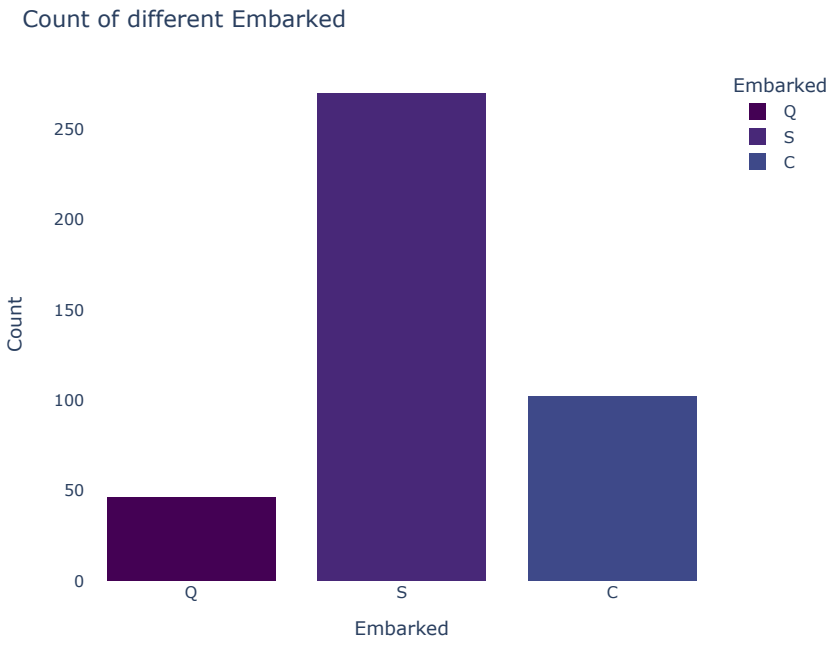

```
fig_age = px.histogram(titanic, x='Age', nbins=30, histnorm='probability density')
fig_age.update_traces(marker=dict(color='#440154'), selector=dict(type='histogram'))
fig_age.update_layout(title='Distribution of Age', title_x=0.5, title_pad=dict(t=20), title_font=dict(size=20), xaxis_title='Age', yaxis
fig_age.show()
```

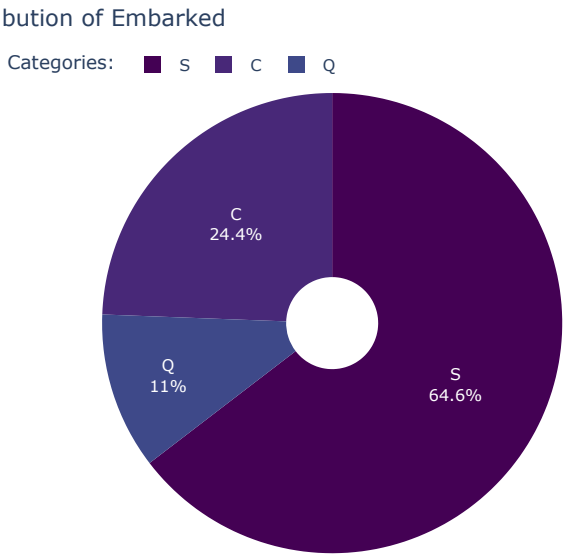



```
fig_fare = px.histogram(titanic, x='Fare', nbins=30, histnorm='probability density')
fig_fare.update_traces(marker=dict(color='#440154'), selector=dict(type='histogram'))
fig_fare.update_layout(title='Distribution of Fare', title_x=0.5, title_pad=dict(t=20), title_font=dict(size=20), xaxis_title='Fare', yax
fig_fare.show()
```

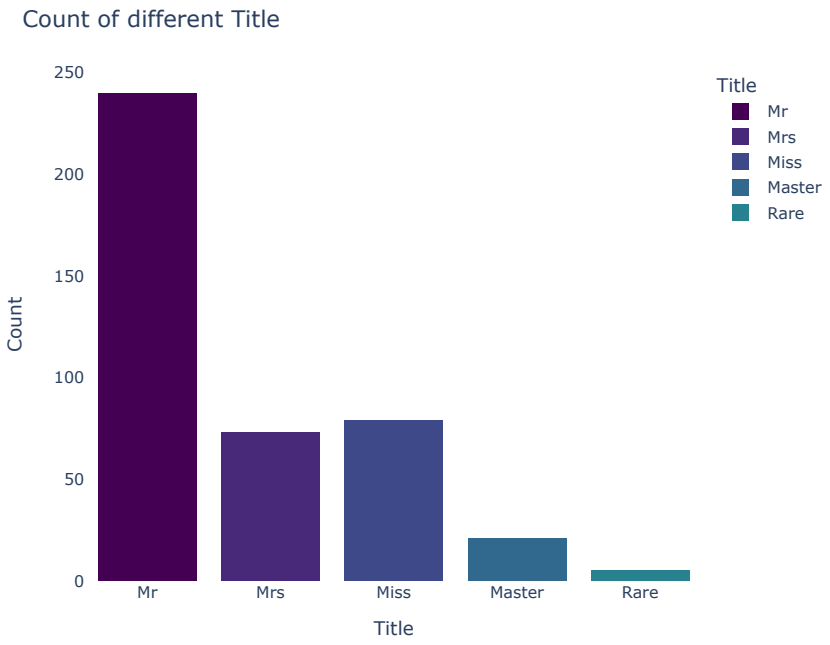


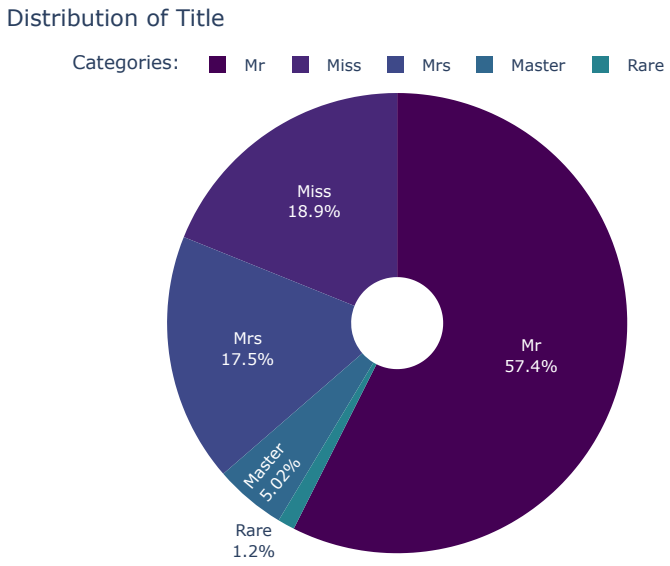

```
fig_embarked_count = px.histogram(titanic, x = 'Embarked', color = 'Embarked', color_discrete_sequence=px.colors.sequential.Viridis)
fig_embarked_count.update_layout(title_text='Count of different Embarked', xaxis_title='Embarked', yaxis_title='Count', plot_bgcolor = 'w
fig_embarked_count.show()

fig_embarked_perc = px.pie(titanic, names= 'Embarked', title=f'Distribution of Embarked', hole=0.2, color_discrete_sequence=px.colors.seq
fig_embarked_perc.update_traces(textinfo='percent+label')
fig_embarked_perc.update_layout(legend_title_text='Categories:', legend=dict(orientation="h", yanchor="bottom", y=1.02))
fig_embarked_perc.show()
```



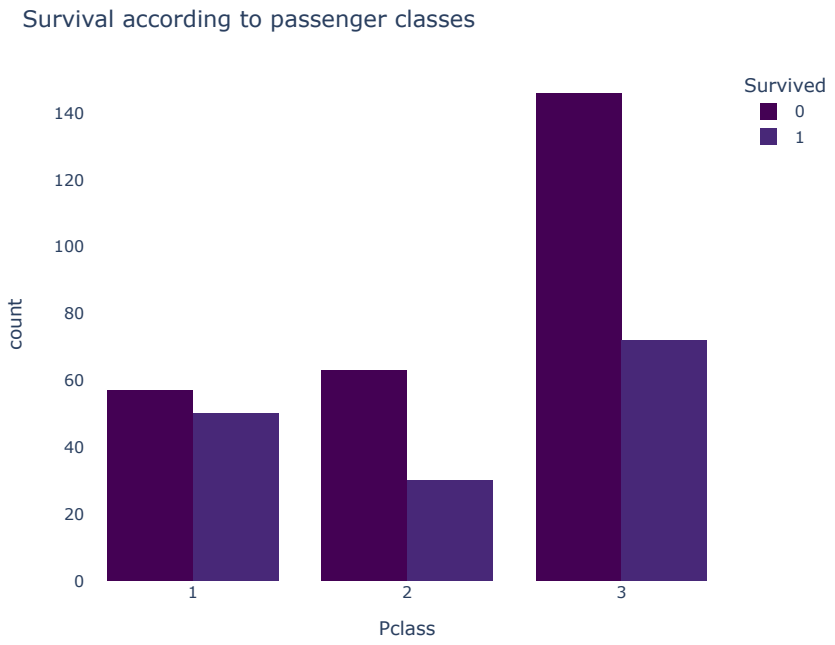



```
fig_title_count = px.histogram(titanic, x = 'Title', color = 'Title', color_discrete_sequence=px.colors.sequential.Viridis)
fig_title_count.update_layout(title_text='Count of different Title', xaxis_title='Title', yaxis_title='Count', plot_bgcolor = 'white')
fig_title_count.show()

fig_title_perc = px.pie(titanic, names= 'Title', title=f'Distribution of Title', hole=0.2, color_discrete_sequence=px.colors.sequential.
fig_title_perc.update_traces(textinfo='percent+label')
fig_title_perc.update_layout(legend_title_text='Categories:', legend=dict(orientation="h", yanchor="bottom", y=1.02))
fig_title_perc.show()
```

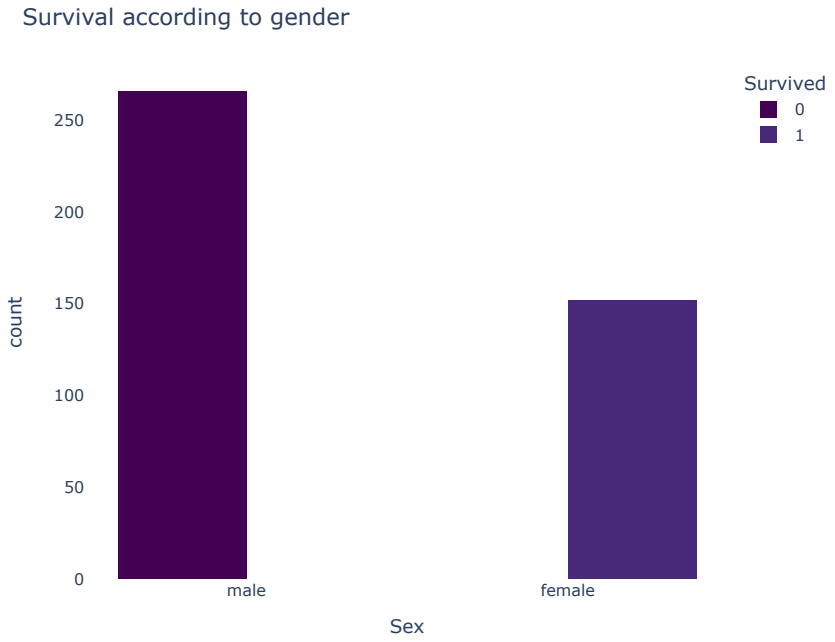




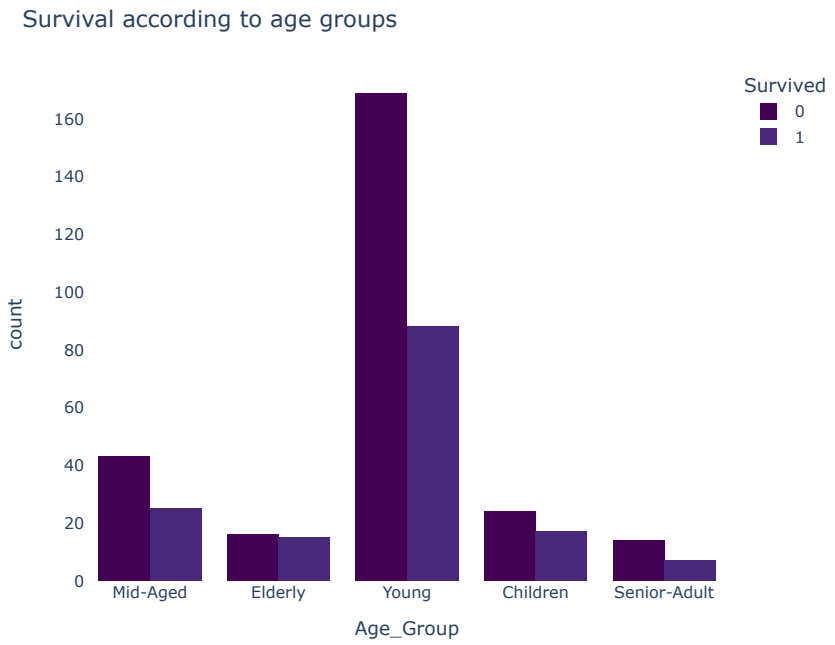

```
fig_pclass_surv = px.histogram(titanic, x = 'Pclass', barmode = 'group', color = 'Survived', color_discrete_sequence=px.colors.sequentia
fig_pclass_surv.update_layout(title = 'Survival according to passenger classes', plot_bgcolor = 'white')
fig_pclass_surv.show()
```



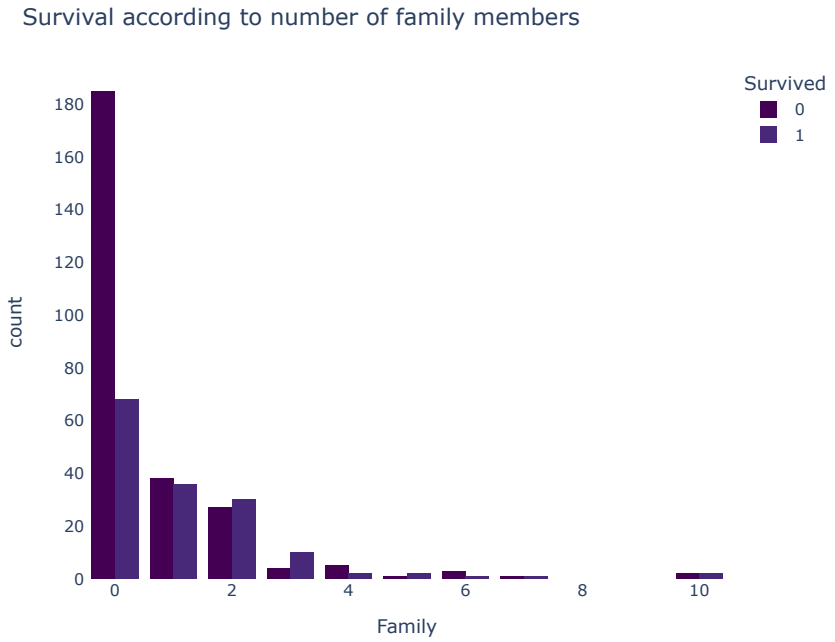

```
fig_pclass_surv = px.histogram(titanic, x = 'Sex', barmode = 'group', color = 'Survived', color_discrete_sequence=px.colors.sequential.\
fig_pclass_surv.update_layout(title = 'Survival according to gender', plot_bgcolor = 'white')
fig_pclass_surv.show()
```




```
fig_embarked_surv = px.histogram(titanic, x = 'Age_Group', barmode = 'group', color = 'Survived', color_discrete_sequence=px.colors.sequ
fig_embarked_surv.update_layout(title = 'Survival according to age groups', plot_bgcolor = 'white')
fig_embarked_surv.show()
```

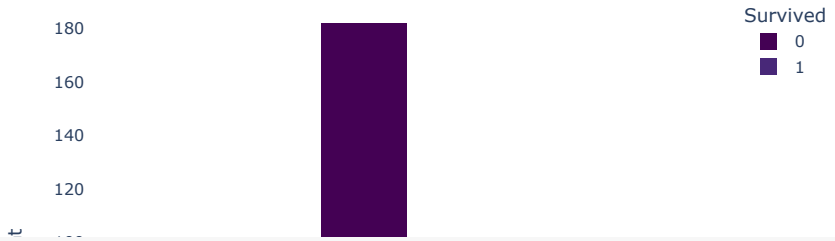



```
fig_family_surv = px.histogram(titanic, x = 'Family', barmode = 'group', color = 'Survived', color_discrete_sequence=px.colors.sequentia
fig_family_surv.update_layout(title = 'Survival according to number of family members', plot_bgcolor = 'white')
fig_family_surv.show()
```




```
fig_embarked_surv = px.histogram(titanic, x = 'Embarked', barmode = 'group', color = 'Survived', color_discrete_sequence=px.colors.seque
fig_embarked_surv.update_layout(title = 'Survival according to embarked', plot_bgcolor = 'white')
fig_embarked_surv.show()
```

Survival according to embarked




```
grouped_data = titanic.groupby(['Age', 'Sex', 'Survived']).agg({'Fare': 'mean'}).reset_index()
fig = px.line(grouped_data, x='Age', y='Fare', color='Survived', facet_col='Sex', facet_col_wrap=2, labels={'Fare': 'Fare', 'Survived':

fig.update_layout(hovermode='x unified', plot_bgcolor = 'white')
fig.update_xaxes(title_text='Age')
fig.update_yaxes(title_text='Fair', row=1, col=1)
fig.show()
```

12. Relation of age and gender with fare