In [ ]:
```
import numpy as np, pandas as pd
import matplotlib.pyplot as plt
```

Loading Dataset

In [ ]:
```
df=pd.read_csv('C:/Users/aunik/Desktop/titanic.csv')
df.shape
```

Out[ ]:
```
(891, 12)
```

In [ ]:
```
df.head()
```

Out[ ]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	I
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	

In [ ]:
```
df['Survived'].value_counts()
```

Out[ ]:
```
0    549
1    342
Name: Survived, dtype: int64
```

549 people have died 342 people have survived

Age Analysis

In [ ]:
```
df['Age'].describe()
```

Out[ ]:
```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

Mapping Age from integer to classes to estimate Child-- Age<15 Middle_Aged-- Age>15 and Age<50 Old-- Age>50

In [ ]:
```
df['Age_Labelled']=df['Age'].apply(lambda x:"Child" if x<15 else ("Middle_Aged" if x>1
```

Created temporary dataframe for plotting purposes

In [ ]:
```
Age_df=df.groupby('Age_Labelled').agg({'Survived':['sum','count','mean']}).reset_index
Age_df.columns=['Age_Type','#Survived','#Travellers','%Survivors']
Age_df
```

Out[ ]:

	Age_Type	#Survived	#Travellers	%Survivors
0	Child	45	78	0.576923
1	Middle_Aged	214	557	0.384201
2	Old	83	256	0.324219

In [ ]:
```
Age_df['#Died']=Age_df['#Travellers']-Age_df['#Survived']
Age_df['%Died']=Age_df['#Died']/Age_df['#Travellers']
Age_df
```

Out[ ]:

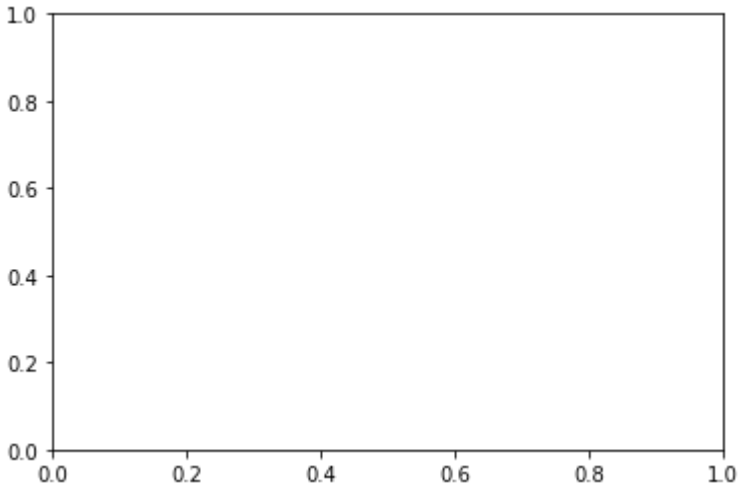
	Age_Type	#Survived	#Travellers	%Survivors	#Died	%Died
0	Child	45	78	0.576923	33	0.423077
1	Middle_Aged	214	557	0.384201	343	0.615799
2	Old	83	256	0.324219	173	0.675781

Density plot and histogram plots to see the frequency of travellers based on Fare

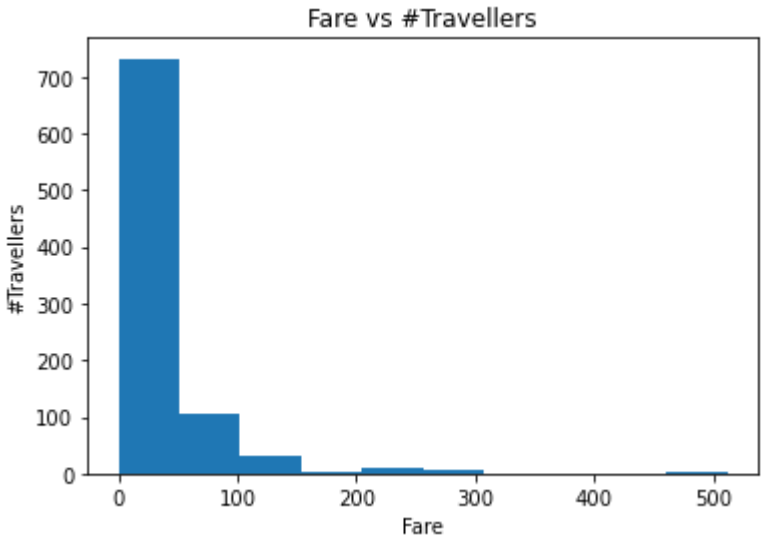
In [ ]:
```
df[['Fare']].plot(kind='density',xlim=[-10,100])
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
c:\Users\aunik\Desktop\Titanic Dataset Analysis-Jupyter Notebook\titanic_jupyternb\Tittanic.ipynb Cell 15' in <cell line: 1>()
----> <a href='vscode-notebook-cell:/c%3A/Users/aunik/Desktop/Titanic%20Dataset%20Analysis-Jupyter%20Notebook/titanic_jupyternb/Tittanic.ipynb#ch0000014?line=0'>1</a> df[['Fare']].plot(kind='density',xlim=[-10,100])

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\plotting\_core.py:972, in PlotAccessor.__call__(self, *args, **kwargs)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_core.py?line=968'>969</a>             label_name = label_kw or data.columns
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_core.py?line=969'>970</a>             data.columns = label_name
--> <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_core.py?line=971'>972</a> return plot_backend.plot(data, kind=kind, **kwargs)

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\plotting\_matplotlib\__init__.py:71, in plot(data, kind, **kwargs)
     <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/__init__.py?line=68'>69</a>         kwargs["ax"] = getattr(ax, "left_ax", ax)
     <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/__init__.py?line=69'>70</a> plot_obj = PLOT_CLASSES[kind](data, **kwargs)
---> <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/__init__.py?line=70'>71</a> plot_obj.generate()
     <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/__init__.py?line=71'>72</a> plot_obj.draw()
     <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/__init__.py?line=72'>73</a> return plot_obj.result

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\plotting\_matplotlib\core.py:329, in MPLPlot.generate(self)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/core.py?line=326'>327</a> self._compute_plot_data()
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/core.py?line=327'>328</a> self._setup_subplots()
--> <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/core.py?line=328'>329</a> self._make_plot()
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/core.py?line=329'>330</a> self._add_table()
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/core.py?line=330'>331</a> self._make_legend()

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\plotting\_matplotlib\hist.py:140, in HistPlot._make_plot(self)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=136'>137</a> if weights is not None and np.ndim(weights) != 1:
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=137'>138</a>     kwds["weights"] = weights[:, i]
```

```
--> <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=139'>140</a> artists = self._plot(ax, y, column_num=i, stacking_id=stacking_id, **kwds)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=141'>142</a> # when by is applied, show title for subplots to know which group it is
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=142'>143</a> if self.by is not None:

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\plotting\_matplotlib\hist.py:213, in KdePlot._plot(cls, ax, y, style, bw_method, ind, column_num, stacking_id, **kwds)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=200'>201</a> @classmethod
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=201'>202</a> def _plot(
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=202'>203</a>     cls,
   (...)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=210'>211</a>     **kwds,
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=211'>212</a> ):
--> <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=212'>213</a>     from scipy.stats import gaussian_kde
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=214'>215</a>     y = remove_na_arraylike(y)
    <a href='file:///c%3A/Users/aunik/AppData/Local/Programs/Python/Python310/lib/site-packages/pandas/plotting/_matplotlib/hist.py?line=215'>216</a>     gkde = gaussian_kde(y, bw_method=bw_method)

ModuleNotFoundError: No module named 'scipy'
```

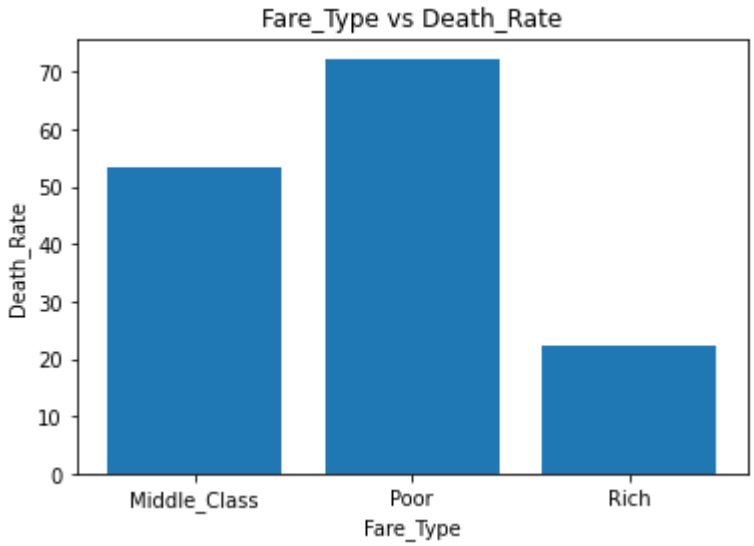


In [ ]:
```
plt.hist(df['Fare'])
plt.title('Fare vs #Travellers')
plt.xlabel("Fare")
plt.ylabel("#Travellers")
plt.show()
```



Bar plot and line plot to see if the curve is monotonic in any sense

In [ ]:
```
plt.bar(Fare_df['Fare_Type'], 100*Fare_df['%Died'],)
plt.title('Fare_Type vs Death_Rate')
plt.xlabel("Fare_Type")
plt.ylabel("Death_Rate")
plt.xticks(x,values)
plt.show()
```



In [ ]:
```
plt.plot(Fare_df['Fare_Type'],Fare_df['%Died'], color ='blue')
plt.title('Fare_Type vs Death_Rate')
plt.xlabel("Fare_Type")
plt.ylabel("Death_Rate")
plt.show()
```

From this, even though there are more number of travellers in middle aged than children the death rate is still high which proves that age is a good differentiator for survival Number of old people travelling is more than children but less than middle aged , but their death rate is the highest. Putting it all together, More the age more was the chance for someone to die.

Sex Analysis

In [ ]:
```
df['Sex'].value_counts()
```

Out[ ]:
```
male      577
female    314
Name: Sex, dtype: int64
```

Created temporary dataframe for plotting purposes

In [ ]:
```
Sex_df=df.groupby('Sex').agg({'Survived':['sum','count','mean']}).reset_index().drople
Sex_df.columns=['Sex','#Survived','#Travellers','%Survivors']
Sex_df
```

Out[ ]:

	Sex	#Survived	#Travellers	%Survivors
0	female	233	314	0.742038
1	male	109	577	0.188908

In [ ]:
```
Sex_df['#Died']=Sex_df['#Travellers']-Sex_df['#Survived']
Sex_df['%Died']=Sex_df['#Died']/Sex_df['#Travellers']
Sex_df
```

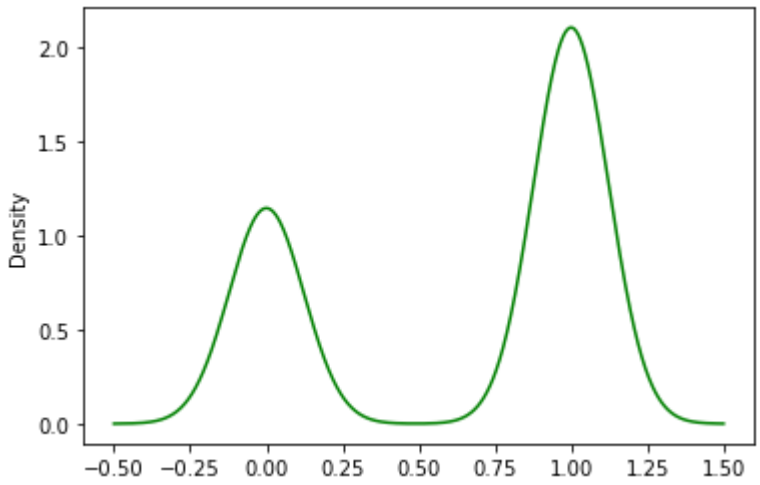
Out[ ]:

	Sex	#Survived	#Travellers	%Survivors	#Died	%Died
0	female	233	314	0.742038	81	0.257962
1	male	109	577	0.188908	468	0.811092

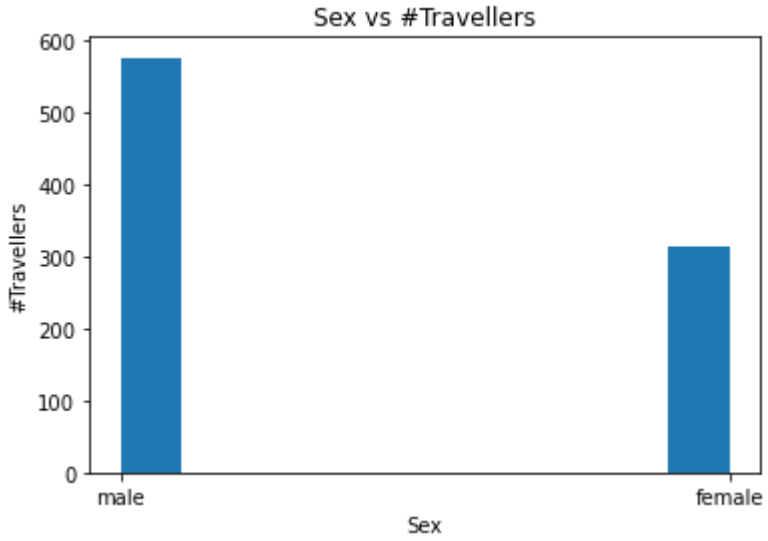
Density plot and histogram plots to see the frequency of travellers based on Sex

In [ ]:
```
df['Sex'].apply(lambda x:1 if x=='male' else 0).plot.density(color='green')

## 1 if male
## 0 if female

##The graph peaks at 0,1 but the peak at 1 is almost twice that of at 0 saying the der
```

Out[ ]:
```
<AxesSubplot:ylabel='Density'>
```



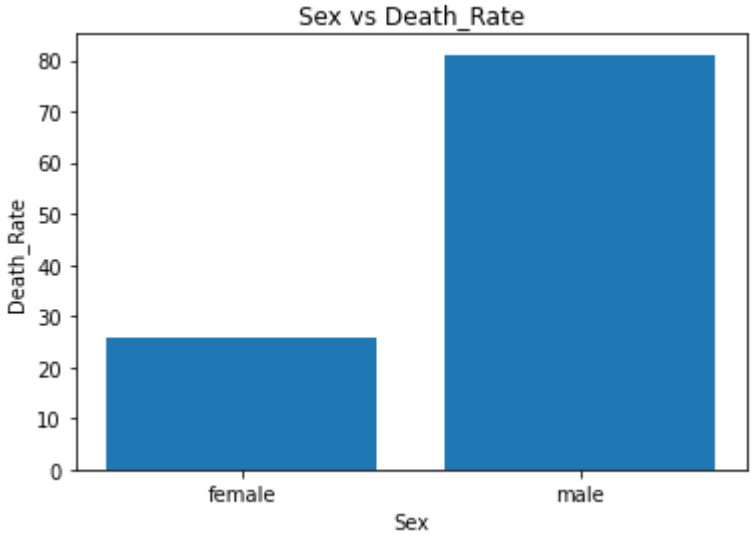
In [ ]:
```
plt.hist(df['Sex'])
plt.title('Sex vs #Travellers')
plt.xlabel("Sex")
plt.ylabel("#Travellers")
plt.show()
```



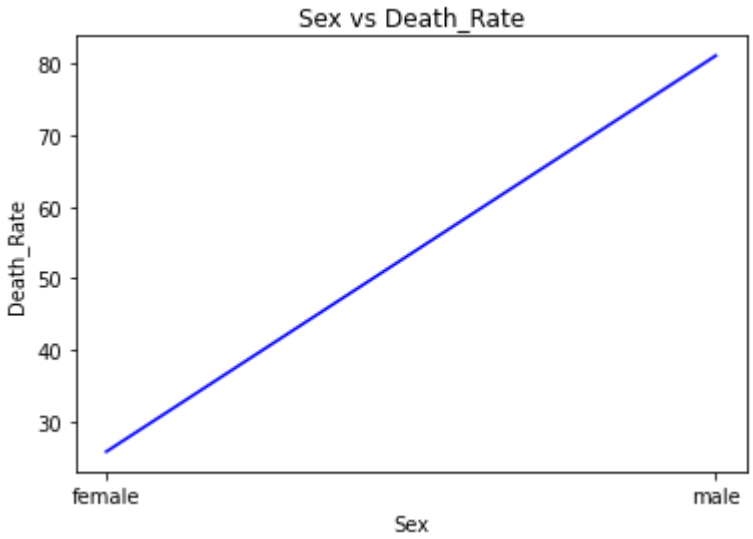
Bar plot and line plot to see if the curve is monotonic in any sense

In [ ]:
```
plt.bar(Sex_df['Sex'], 100*Sex_df['%Died'])
```

```
plt.title('Sex vs Death_Rate')
plt.xlabel("Sex")
plt.ylabel("Death_Rate")
plt.show()
```



In [ ]:
```
plt.plot(Sex_df['Sex'],100*Sex_df['%Died'], color ='blue')
plt.title('Sex vs Death_Rate')
plt.xlabel("Sex")
plt.ylabel("Death_Rate")
plt.show()
```



From this, even though there are more number of male travellers than female traveller(Almost double) the death rate is very high almost 80% in male when compared to female (25%) which proves that the casulaty rate was more in male and "SEX" can be used as a a very strong indicator to identify Survival Rate

Financial Status Analysis

As we don't have a clear defined rich/poor segregation we will be using the fare price as a proxy to identify one's financial capability

In [ ]:
```
df['Fare'].quantile([0.1,0.3,0.5,0.7,0.9])
```

Out[ ]:
```
0.1     7.5500
0.3     8.0500
0.5    14.4542
0.7    27.0000
0.9    77.9583
Name: Fare, dtype: float64
```

In [ ]:
```
df['Fare'].mean()
```

Out[ ]:
```
32.2042079685746
```

50% of the travellers are travelling within the far of 14.4542$whereastheavergefarepricewas32.204$ This says that there are more travellers travelling with lower ticket fares but only about 10% of travellers have paid a much higher price which made the mean value so high To prove this the 90% quantile value says us that 90% of the travellers have paid only 77.95$orlessbutthehighestpricewasabout500$Mapping Fare from continous values to discrete classes Rich-- Fare>80$Middle_Class--Fare>20$ and Fare<80$Poor--Fare<20$

In [ ]:
```
df['Fare_Labelled']=df['Fare'].apply(lambda x:"Poor" if x<20 else ("Middle_Class" if x
```

In [ ]:
```
df['Fare_Labelled'].value_counts()
```

Out[ ]:
```
Poor            515
Middle_Class    300
Rich             76
Name: Fare_Labelled, dtype: int64
```

Created temporary dataframe for plotting purposes

In [ ]:
```
Fare_df=df.groupby('Fare_Labelled').agg({'Survived':['sum','count','mean']}).reset_inc
Fare_df.columns=['Fare_Type','#Survived','#Travellers','%Survivors']
Fare_df
```

Out[ ]:

	Fare_Type	#Survived	#Travellers	%Survivors
0	Middle_Class	140	300	0.466667
1	Poor	143	515	0.277670
2	Rich	59	76	0.776316

In [ ]:
```
Fare_df['#Died']=Fare_df['#Travellers']-Fare_df['#Survived']
Fare_df['%Died']=Fare_df['#Died']/Fare_df['#Travellers']
Fare_df
```

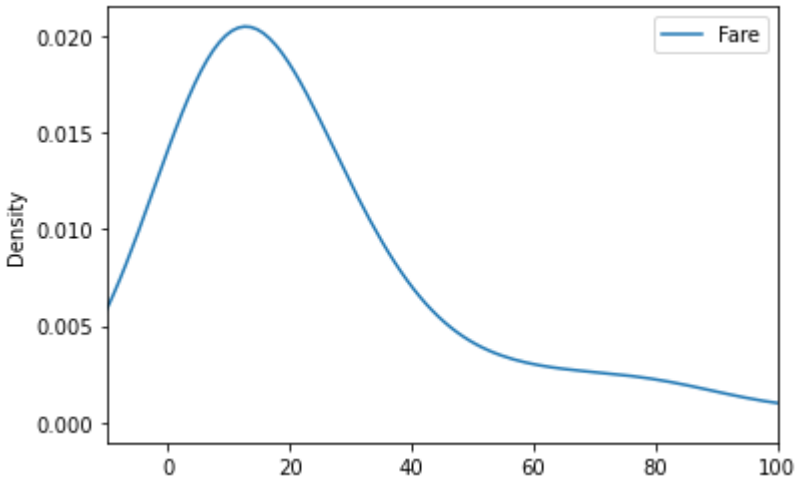
Out[ ]:

	Fare_Type	#Survived	#Travellers	%Survivors	#Died	%Died
0	Middle_Class	140	300	0.466667	160	0.533333
1	Poor	143	515	0.277670	372	0.722330
2	Rich	59	76	0.776316	17	0.223684

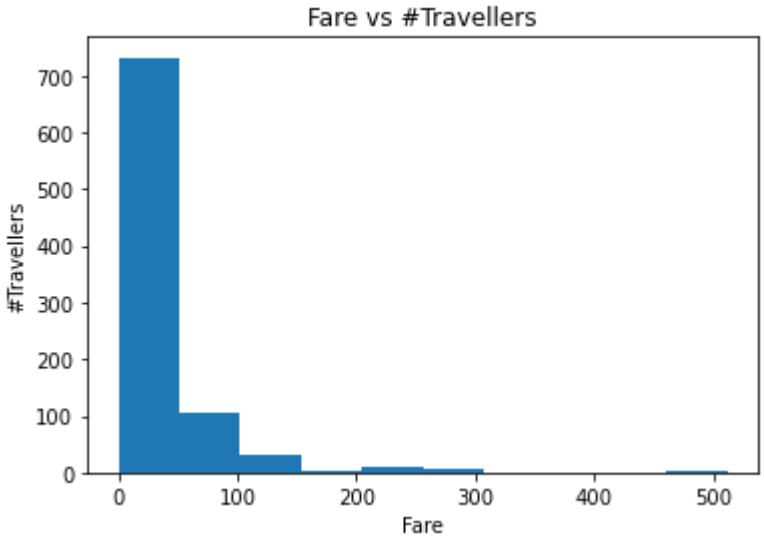
Density plot and histogram plots to see the frequency of travellers based on Age

In [ ]:
```
df[['Fare']].plot(kind='density',xlim=[-10,100])
```

Out[ ]:
```
<AxesSubplot:ylabel='Density'>
```

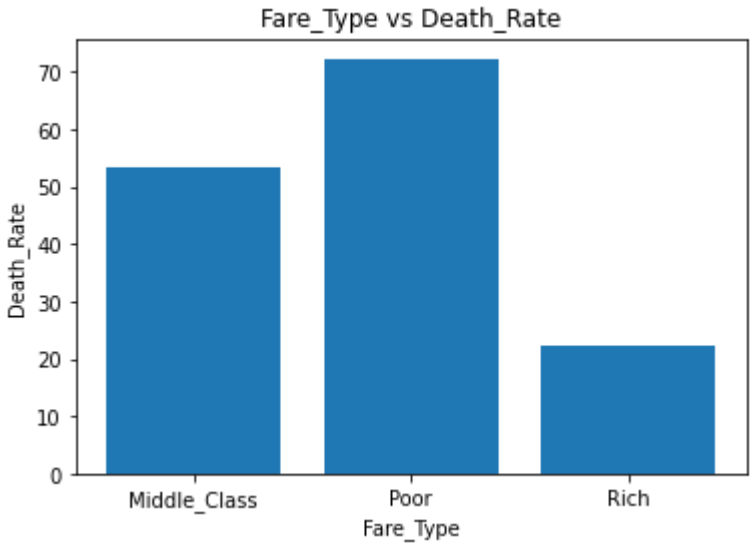


In [ ]:
```
plt.hist(df['Fare'])
plt.title('Fare vs #Travellers')
plt.xlabel("Fare")
plt.ylabel("#Travellers")
plt.show()
```

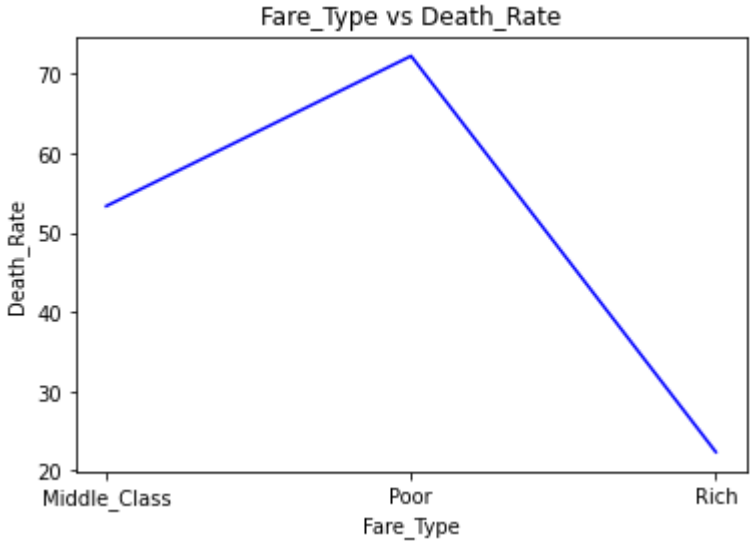


Bar plot and line plot to see if the curve is monotonic in any sense

In [ ]:
```
plt.bar(Fare_df['Fare_Type'], 100*Fare_df['%Died'])
plt.title('Fare_Type vs Death_Rate')
plt.xlabel("Fare_Type")
plt.ylabel("Death_Rate")
plt.show()
```



In [ ]:
```
plt.plot(Fare_df['Fare_Type'],100*Fare_df['%Died'], color ='blue')
plt.title('Fare_Type vs Death_Rate')
plt.xlabel("Fare_Type")
plt.ylabel("Death_Rate")
plt.show()
```



From this, even though there are more number of travellers who are poor as compared to middle class or rich the death rates are significantly higher there. The number of middle class travellers is more than rich people but their death rate is comparibly higher than the rich This clearly shows us the more the price a traveller has paid the more was his chance for survival. "Fare" is a strong indicator for survival rate.

Conclusion

All three "Fare","Age","Gender" are very strong indicators for predicting if someone could survive the titanic.

In [ ]: