Titanic: Machine Learning from Disaster ¶

Data Fields

- **Survival** - Survival. 0 = No, 1 = Yes
- **Pclass** - Ticket class. 1 = 1st, 2 = 2nd, 3 = 3rd
- **Sex** - Sex.
- **Age** - Age in years.
- **SibSp** - # of siblings / spouses aboard the Titanic.
- **Parch** - # of parents / children aboard the Titanic.
- **Ticket** - Ticket number.
- **Fare** - Passenger fare.
- **Cabin** - Cabin number.
- **Embarked** - Port of Embarkation. C = Cherbourg, Q = Queenstown, S = Southampton

01. 데이터 불러오기

In [46]:

```
import matplotlib
import matplotlib.pylab as pylab
import matplotlib.pyplot as plt
import matplotlib as mpl
import seaborn as sns

import pandas as pd
import numpy as np

import xgboost as xgb
import sklearn
import warnings

from sklearn.metrics import make_scorer, accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
import scipy
import numpy
import json
import sys
import csv
import os
```

In [47]:

```
# import train and test to play with it
df_train = pd.read_csv('data/train.csv')
df_test = pd.read_csv('data/test.csv')
```

In [48]:

```
print( type(df_train), type(df_test) )
```

```
<class 'pandas.core.frame.DataFrame'> <class 'pandas.core.frame.DataFrame'>
```

1-2 버전 확인

In [50]:

```
print('matplotlib: {}'.format(matplotlib.__version__))
print('sklearn: {}'.format(sklearn.__version__))
print('scipy: {}'.format(scipy.__version__))
print('seaborn: {}'.format(sns.__version__))
print('pandas: {}'.format(pd.__version__))
print('numpy: {}'.format(np.__version__))
print('Python: {}'.format(sys.version))
```

```
matplotlib: 2.1.2
sklearn: 0.19.1
scipy: 1.0.0
seaborn: 0.8.1
pandas: 0.22.0
numpy: 1.14.0
Python: 3.6.4 |Anaconda, Inc.| (default, Jan 16 2018, 10:22:32) [MSC v.1900 64 bit (AMD64)]
```

In [51]:

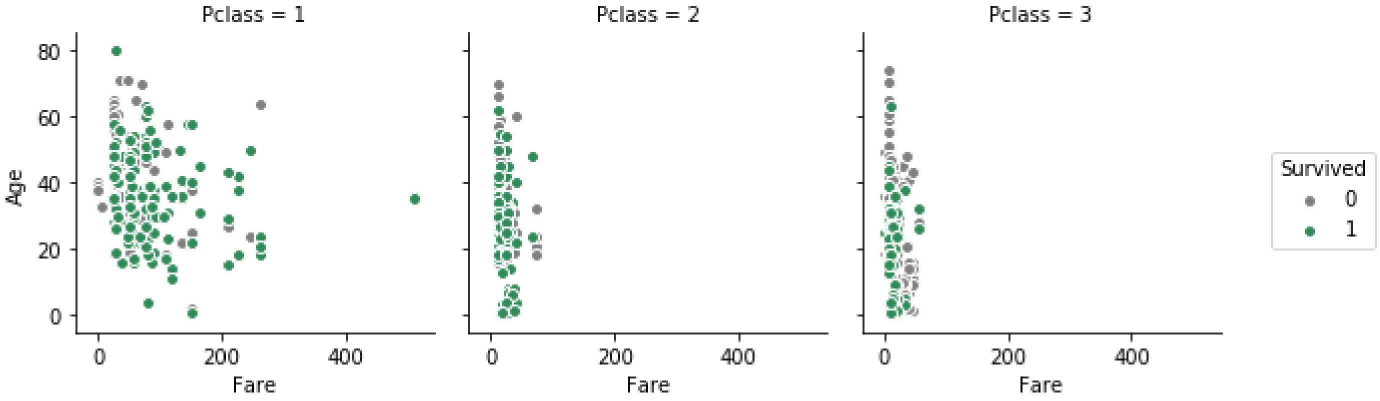
```
sns.set(style='white', context='notebook', palette='deep')
pylab.rcParams['figure.figsize'] = 12,8
warnings.filterwarnings('ignore')
mpl.style.use('ggplot')
sns.set_style('white')
%matplotlib inline
```

02. EDA

2-1 Scatter plot(산점도)

- 두 양적 변수간의 관계를 확인 목적을 갖습니다.

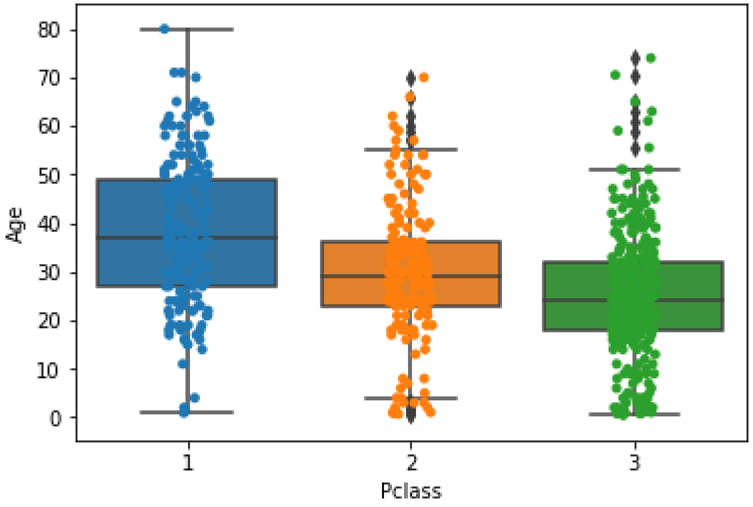
In [8]:
```
# Modify the graph above by assigning each species an individual color.
g = sns.FacetGrid(df_train, hue="Survived", col="Pclass", margin_titles=True,
                  palette={1:"seagreen", 0:"gray"})
g=g.map(plt.scatter, "Fare", "Age",edgecolor="w").add_legend();
```



2-2 BoxPlot(상자 그림)

- 상자 그림은 사분위수를 통해 수치 데이터 그룹을 그래픽으로 묘사합니다.
- 이상치와 75%, 중앙값 25%의 값과 분포를 확인할 수 있습니다.

In [12]:
```
ax= sns.boxplot(x="Pclass", y="Age", data=df_train)
ax= sns.stripplot(x="Pclass", y="Age", data=df_train, jitter=True, edgecolor="gray")
plt.show()
```



2-3 Histogram(히스토그램)

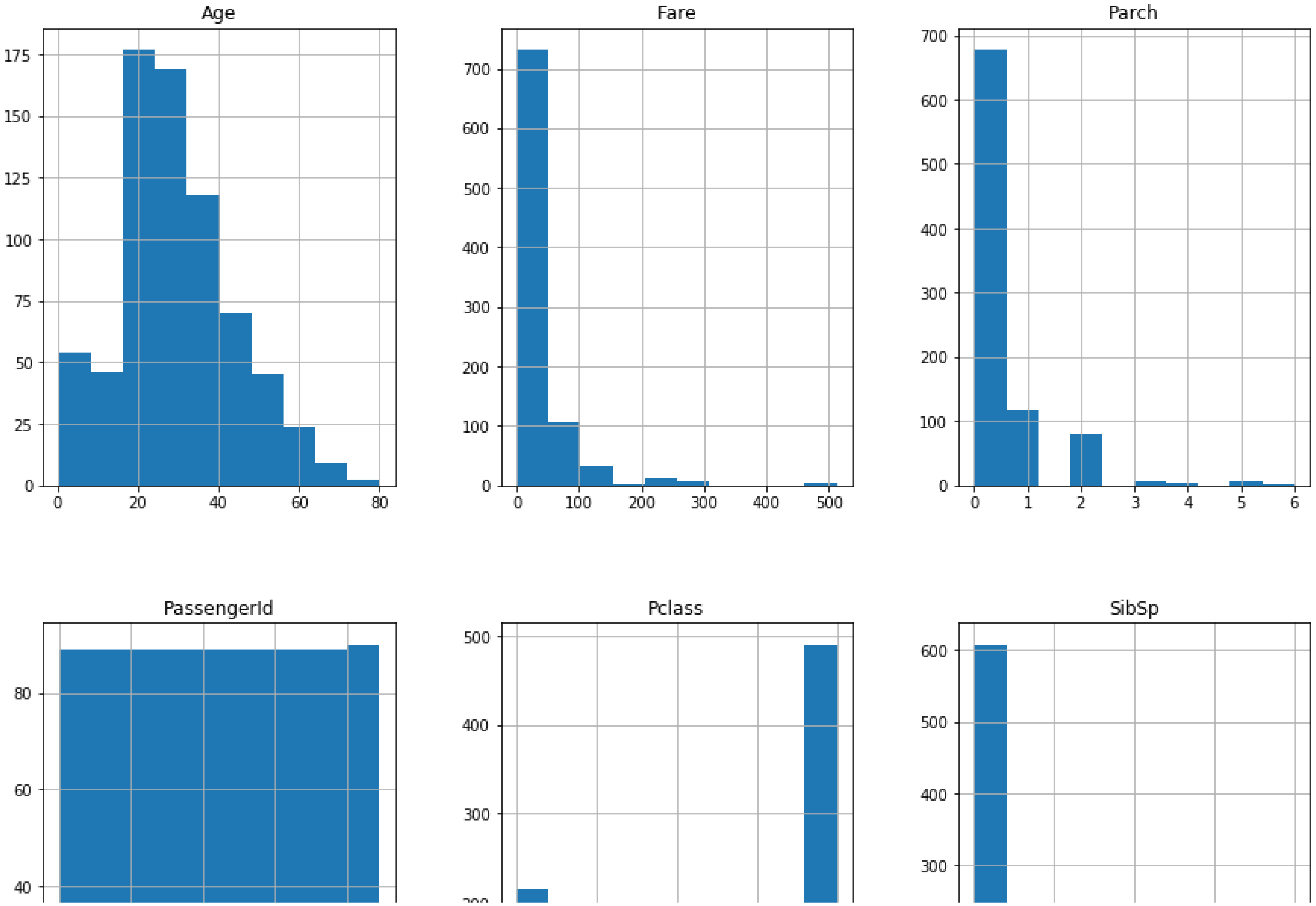
- 각각의 입력 변수에 대한 분포를 확인할 수 있습니다.

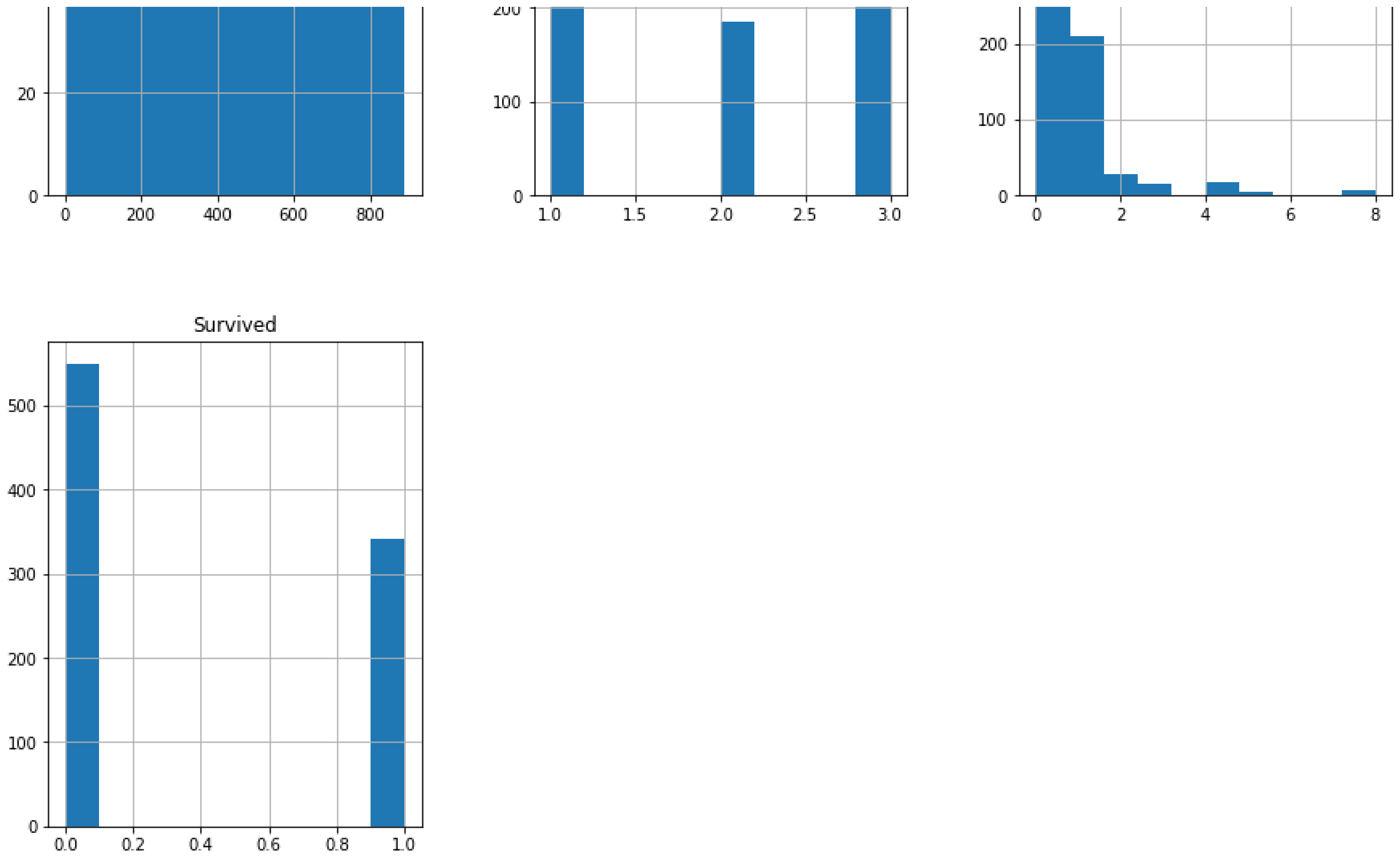
In [13]:

```
# histograms
df_train.hist(figsize=(15,20))
plt.figure()
```

Out[13]:

```
<matplotlib.figure.Figure at 0x12dfbd55710>
```



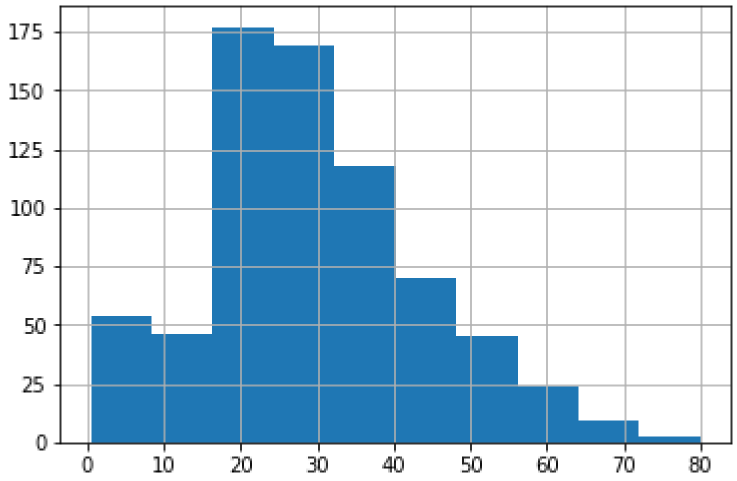


```
<matplotlib.figure.Figure at 0x12dfbd55710>
```

- Age가 가우시안 분포를 갖는 것 같습니다.

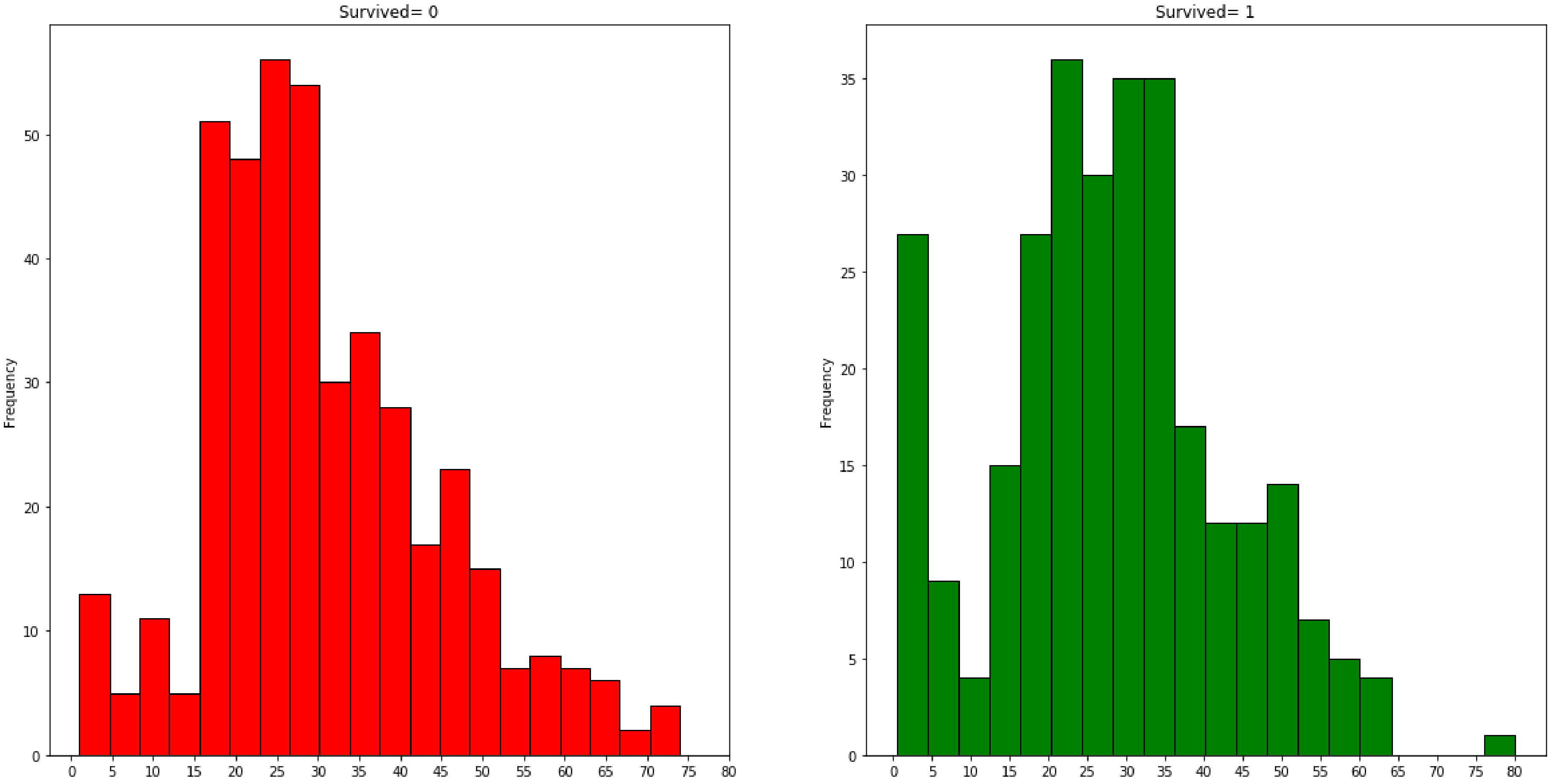
In [14]:

```
df_train["Age"].hist();
```



In [17]:

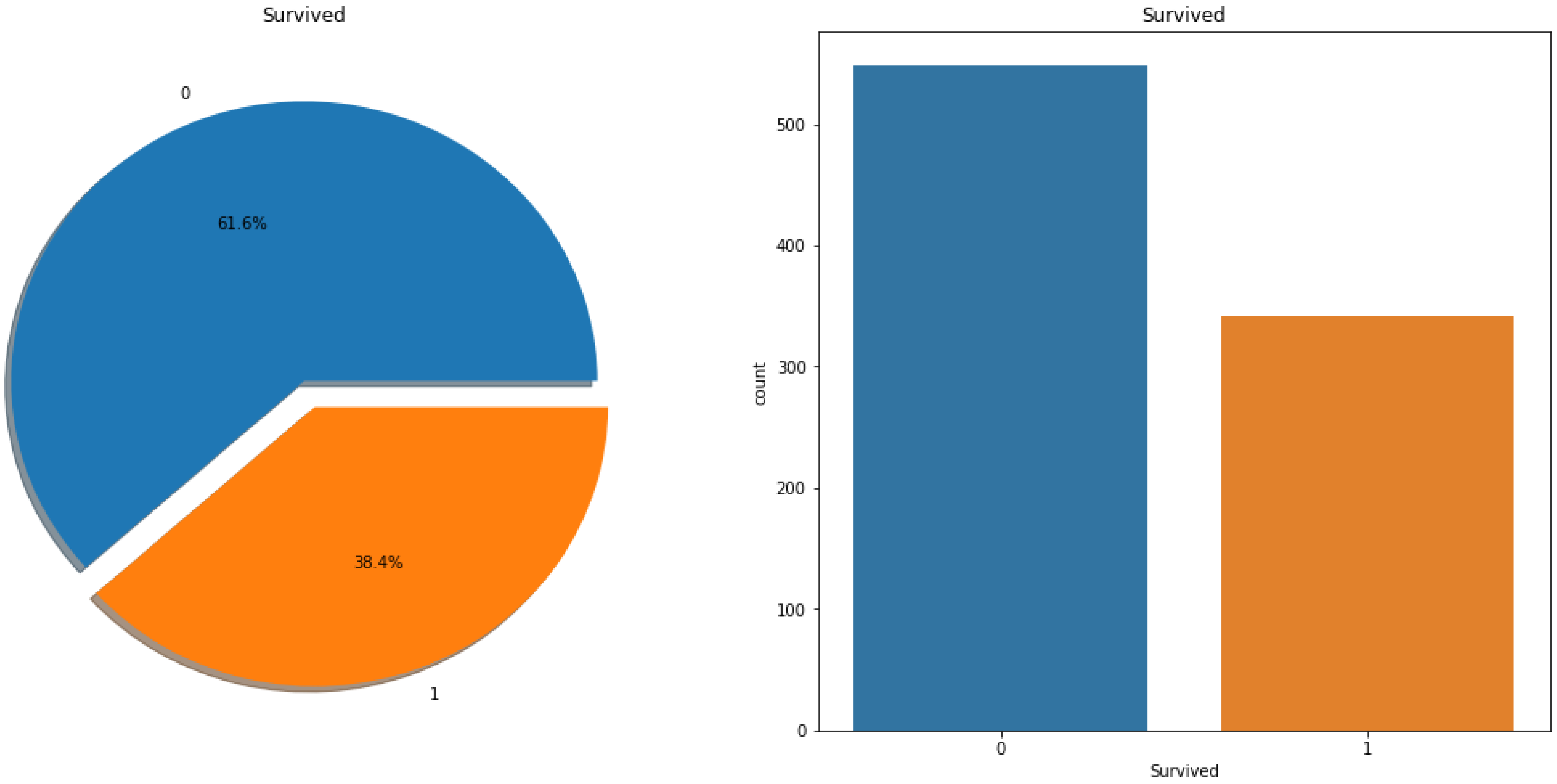
```
f,ax=plt.subplots(1,2,figsize=(20,10))
df_train[df_train['Survived']==0].Age.plot.hist(ax=ax[0],
                                        bins=20,edgecolor='black',color='red')

ax[0].set_title('Survived= 0')
x1=list(range(0,85,5))
ax[0].set_xticks(x1)     # 첫번째 그래프 x축 눈금
df_train[df_train['Survived']==1].Age.plot.hist(ax=ax[1],
                                        bins=20,edgecolor='black', color='green')

ax[1].set_title('Survived= 1')
x2=list(range(0,85,5))
ax[1].set_xticks(x2)     # 두번째 그래프 x축 눈금
plt.show()
```


Survived= 0
Frequency
0
10
20
30
40
50
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
Survived= 1
Frequency
0
5
10
15
20
25
30
35
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80

pie 그래프

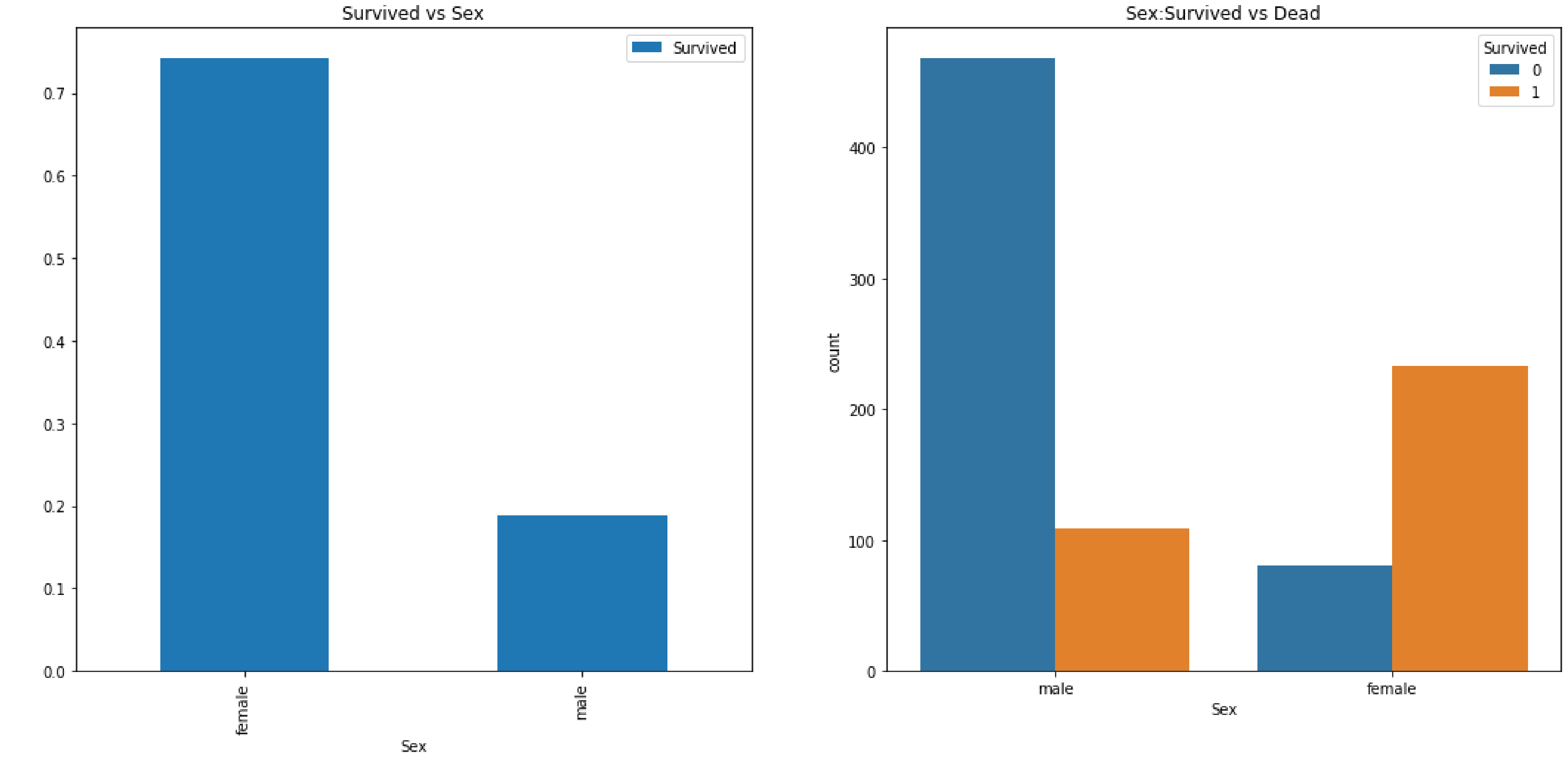
In [19]:

```
f,ax=plt.subplots(1,2,figsize=(18,8))
df_train['Survived'].value_counts().plot.pie(explode=[0,0.1],
                                             autopct='%1.1f%%',ax=ax[0],shadow=True)
ax[0].set_title('Survived')
ax[0].set_ylabel('')

sns.countplot('Survived',data=df_train,ax=ax[1])
ax[1].set_title('Survived')
plt.show()
```



In [21]:

```
f,ax=plt.subplots(1,2,figsize=(18,8))

# 첫번째 그래프
df_train[['Sex','Survived']].groupby(['Sex']).mean().plot.bar(ax=ax[0])
ax[0].set_title('Survived vs Sex')

# 두번째 그래프
sns.countplot('Sex',hue='Survived',data=df_train,ax=ax[1])
ax[1].set_title('Sex:Survived vs Dead')
plt.show()
```



2-4 Multivariate Plots(다변량 플롯)

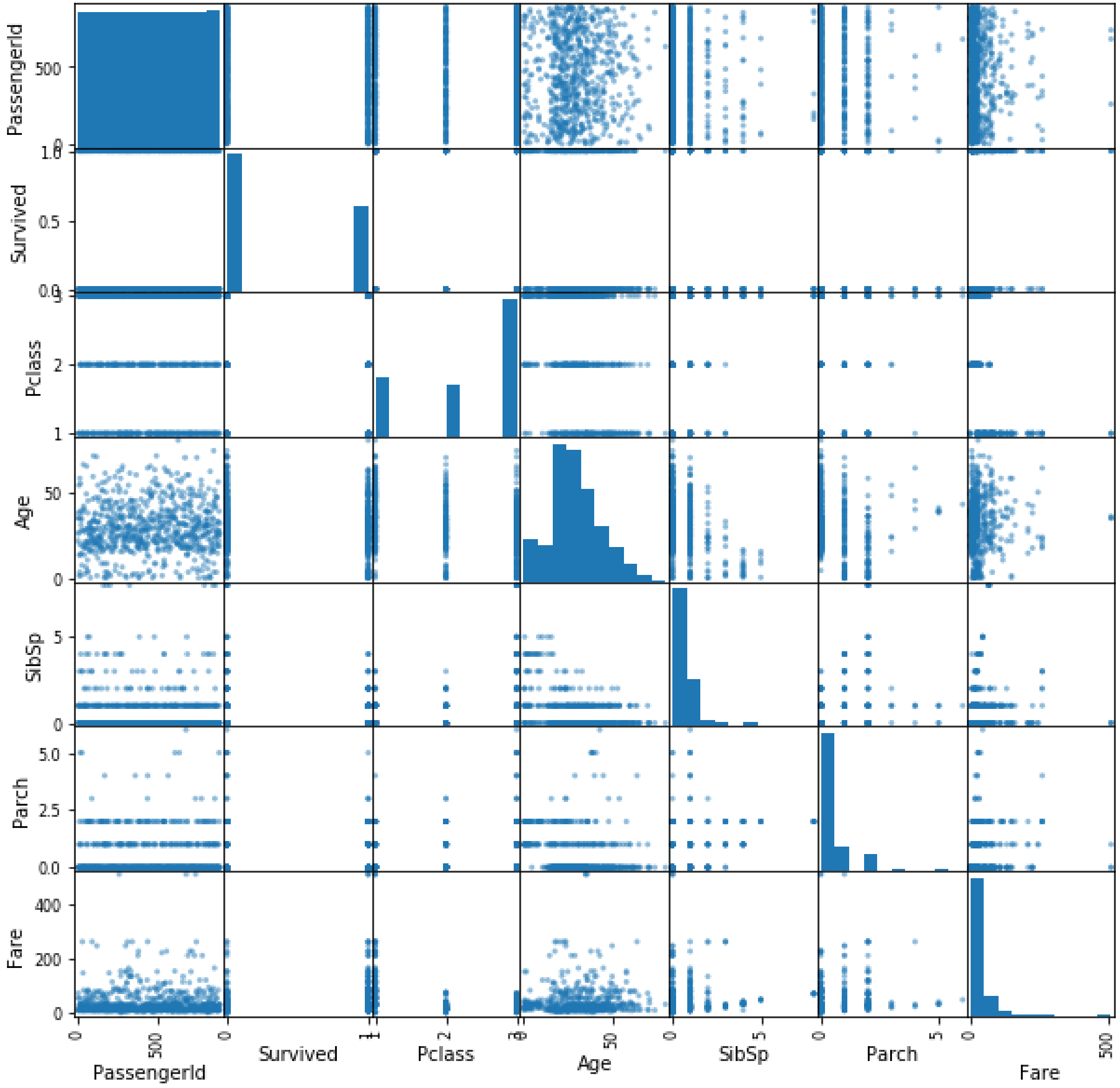
- 모든 속성 쌍의 산점도를 확인해 볼 수 있다.
- 입력 변수간의 구조화된 관계를 발견하는 데 도움이 될 수 있다.

In [22]:

```
# scatter plot matrix
pd.plotting.scatter_matrix(df_train,figsize=(10,10))
plt.figure()
```

Out[22]:

```
<matplotlib.figure.Figure at 0x12dfc66c358>
```



```
<matplotlib.figure.Figure at 0x12dfc66c358>
```

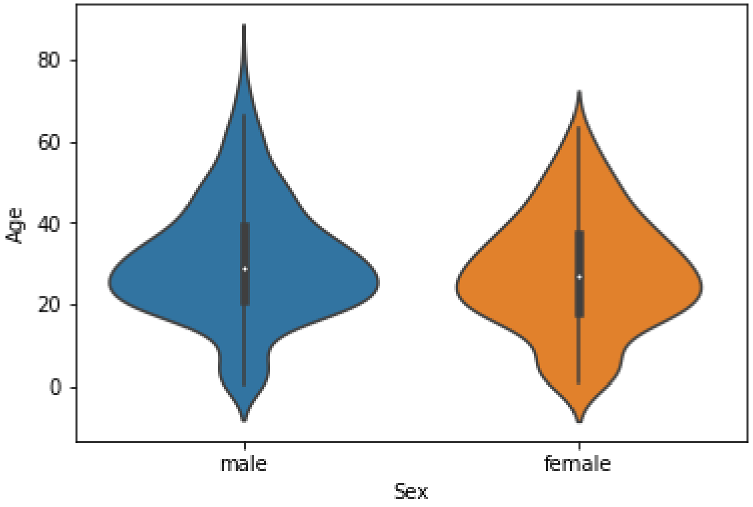
2-5 violinplots

In [23]:

```
sns.violinplot(data=df_train,x="Sex", y="Age")
```

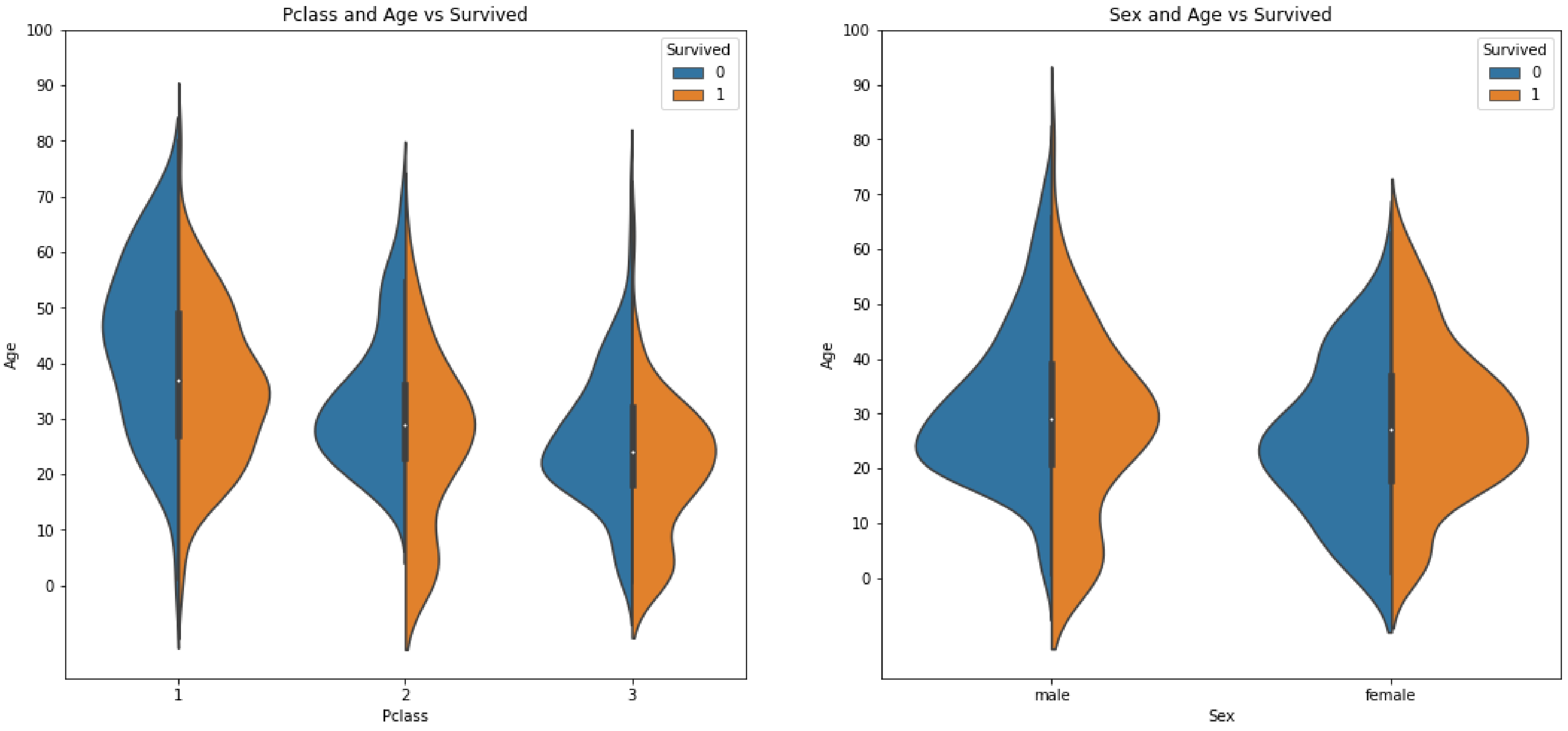
Out[23]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x12dfbc3b128>
```



In [24]:

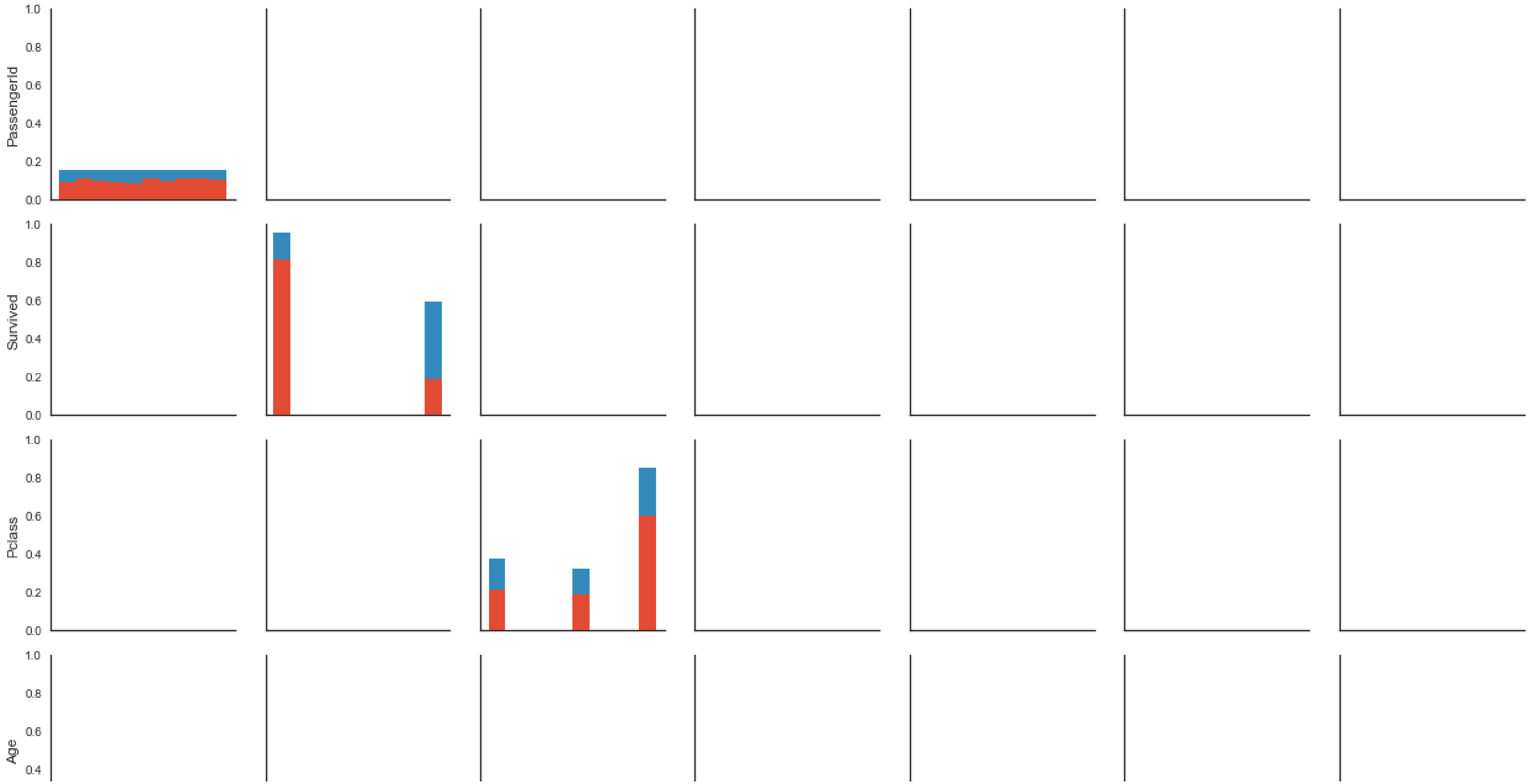
```
f,ax=plt.subplots(1,2,figsize=(18,8))

### 첫번째 그래프
sns.violinplot("Pclass","Age", hue="Survived", data=df_train,split=True,ax=ax[0])
ax[0].set_title('Pclass and Age vs Survived')
ax[0].set_yticks(range(0,110,10))

### 두번째 그래프
sns.violinplot("Sex","Age", hue="Survived", data=df_train,split=True,ax=ax[1])
ax[1].set_title('Sex and Age vs Survived')
ax[1].set_yticks(range(0,110,10))
plt.show()
```

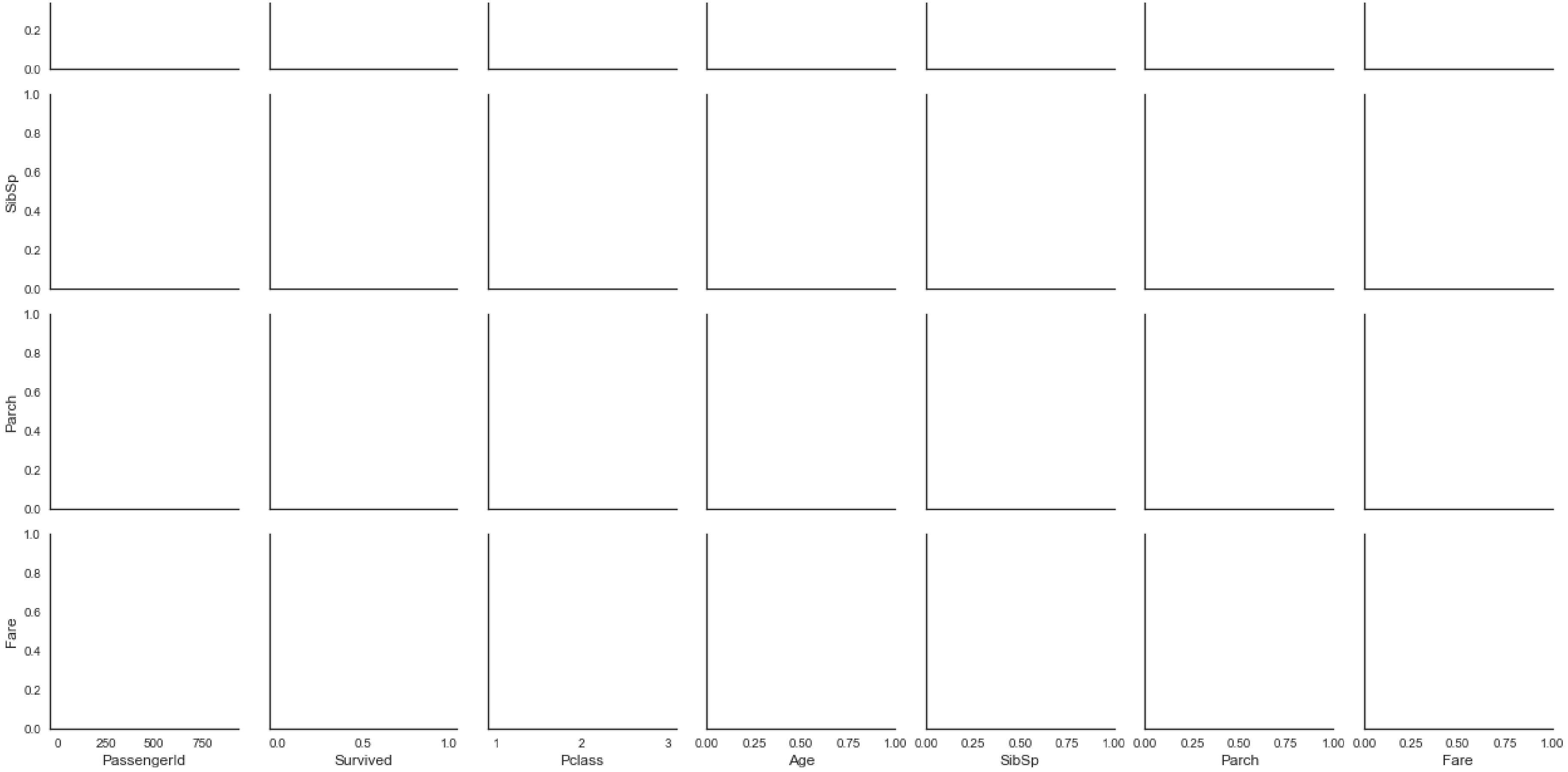


2-6 pairplot

In [52]:
```
# Using seaborn pairplot to see the bivariate relation between each pair of features
sns.pairplot(df_train, hue="Sex")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-52-f2fc604c1cc4> in <module>()
      1 # Using seaborn pairplot to see the bivariate relation between each pair of features
----> 2 sns.pairplot(df_train, hue="Sex")

~₩Anaconda3₩lib₩site-packages₩seaborn₩axisgrid.py in pairplot(data, hue, hue_order, palette, vars, x_vars, y_vars, kind, diag_
kind, markers, size, aspect, dropna, plot_kws, diag_kws, grid_kws)
   2058     if grid.square_grid:
   2059         if diag_kind == "hist":
-> 2060             grid.map_diag(plt.hist, **diag_kws)
   2061         elif diag_kind == "kde":
   2062             diag_kws["legend"] = False

~₩Anaconda3₩lib₩site-packages₩seaborn₩axisgrid.py in map_diag(self, func, **kwargs)
   1363                     func(vals, color=color, **kwargs)
   1364                 else:
-> 1365                     func(vals, color=color, histtype="barstacked", **kwargs)
   1366 
   1367             else:

~₩Anaconda3₩lib₩site-packages₩matplotlib₩pyplot.py in hist(x, bins, range, density, weights, cumulative, bottom, histtype, ali
gn, orientation, rwidth, log, color, label, stacked, normed, hold, data, **kwargs)
   3023                       histtype=histtype, align=align, orientation=orientation,
   3024                       rwidth=rwidth, log=log, color=color, label=label,
-> 3025                       stacked=stacked, normed=normed, data=data, **kwargs)
   3026     finally:
   3027         ax._hold = washold

~₩Anaconda3₩lib₩site-packages₩matplotlib₩__init__.py in inner(ax, *args, **kwargs)
   1715                     warnings.warn(msg % (label_namer, func.__name__),
   1716                                   RuntimeWarning, stacklevel=2)
-> 1717             return func(ax, *args, **kwargs)
   1718         pre_doc = inner.__doc__
   1719         if pre_doc is None:

~₩Anaconda3₩lib₩site-packages₩matplotlib₩axes₩_axes.py in hist(***failed resolving arguments***)
   6163             # this will automatically overwrite bins,
   6164             # so that each histogram uses the same bins
-> 6165             m, bins = np.histogram(x[i], bins, weights=w[i], **hist_kwargs)
   6166             m = m.astype(float)  # causes problems later if it's an int
   6167             if mlast is None:

~₩Anaconda3₩lib₩site-packages₩numpy₩lib₩function_base.py in histogram(a, bins, range, normed, weights, density)
    665     if first_edge > last_edge:
    666         raise ValueError(
--> 667             'max must be larger than min in range parameter.')
    668     if not np.all(np.isfinite([first_edge, last_edge])):
    669         raise ValueError(

ValueError: max must be larger than min in range parameter.
```



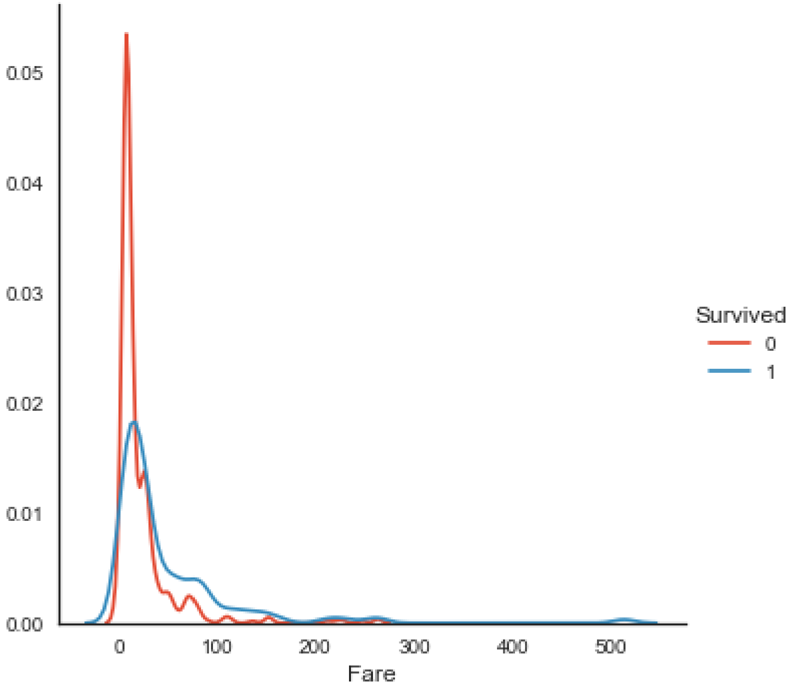


2-7 kdeplot

- pairplot의 막대그래프의 대각선에 표시된 막대 그래프를 kde로 대체 가능합니다.

In [54]:

```python
# seaborn's kdeplot, plots univariate or bivariate density estimates.
#Size can be changed by tweeking the value used
sns.FacetGrid(df_train, hue="Survived", size=5).map(sns.kdeplot, "Fare").add_legend()
plt.show()
```



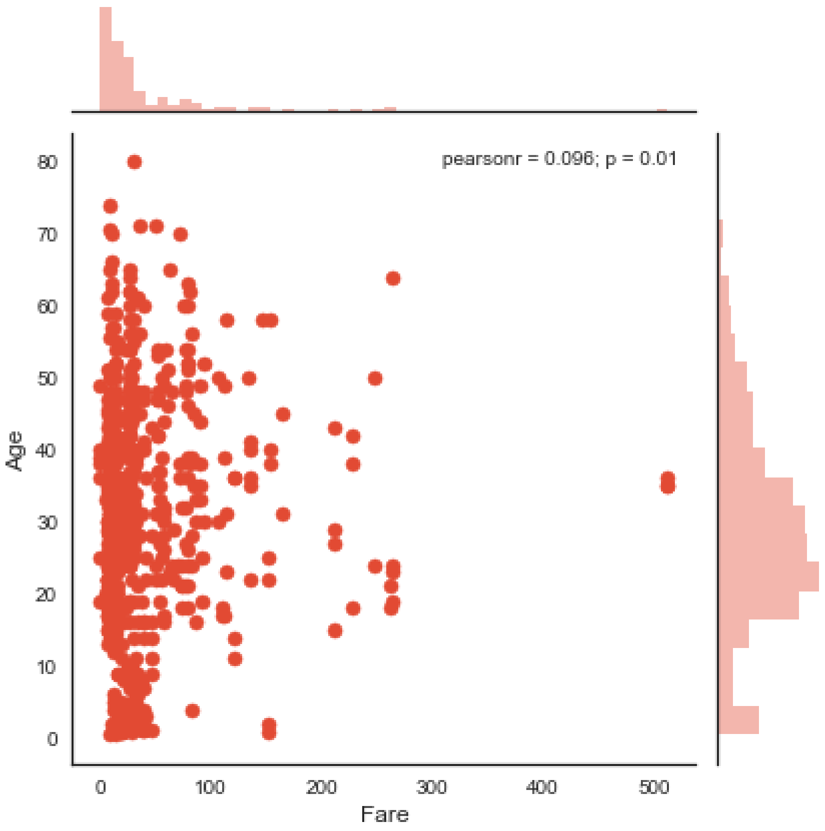
2-8 jointplot

In [55]:

```
sns.jointplot(x='Fare',y='Age',data=df_train)
```

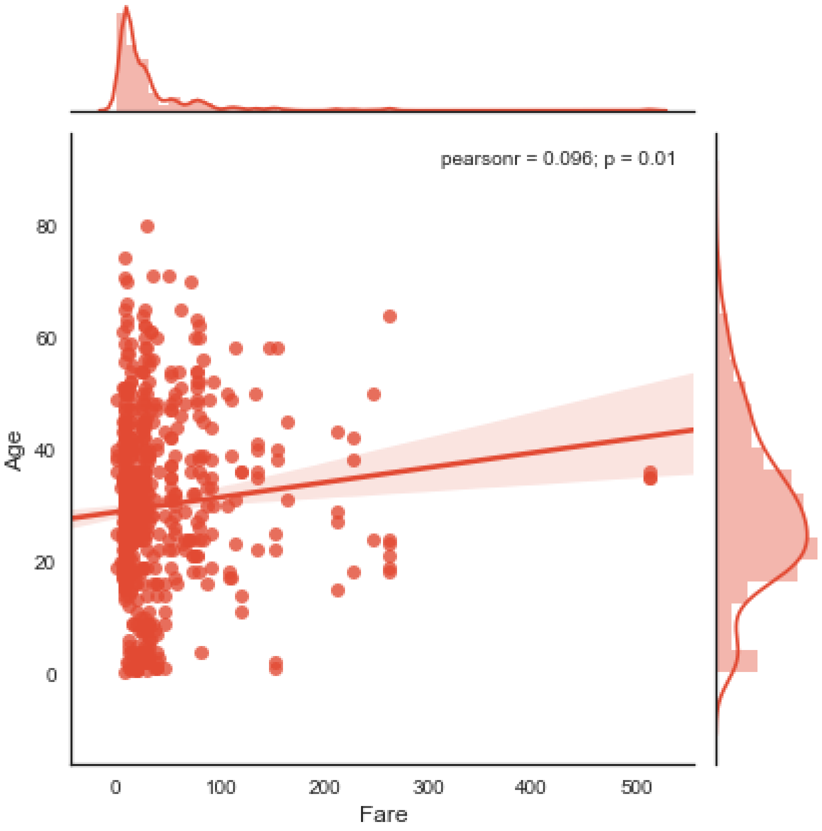
Out[55]:

```
<seaborn.axisgrid.JointGrid at 0x12dfc951eb8>
```



In [56]:
```
sns.jointplot(x='Fare',y='Age' ,data=df_train, kind='reg')
```

Out[56]:
```
<seaborn.axisgrid.JointGrid at 0x12dfc8a6ba8>
```



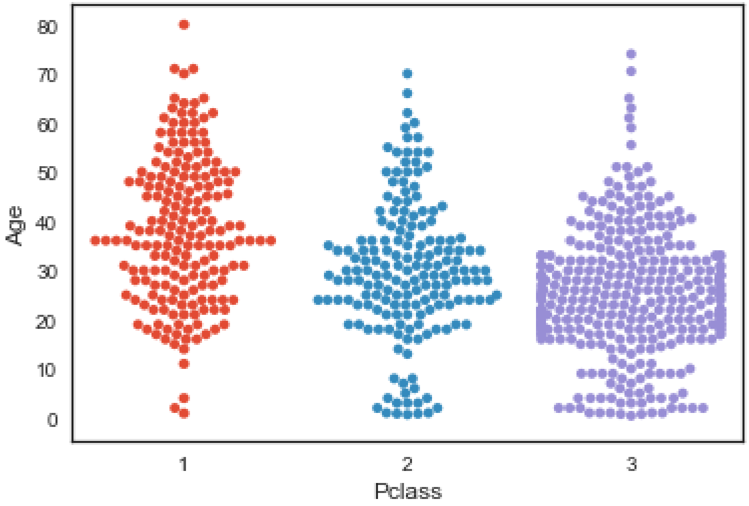
2-9 Swarm plot

In [57]:

```
sns.swarmplot(x='Pclass',y='Age',data=df_train)
```

Out[57]:

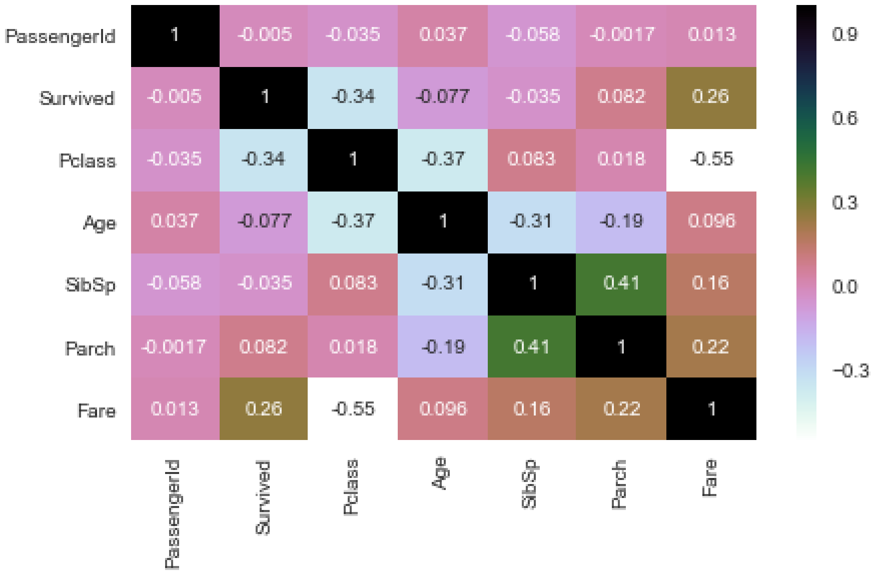
```
<matplotlib.axes._subplots.AxesSubplot at 0x12dfc0f4be0>
```



2-10 Heatmap

In [59]:

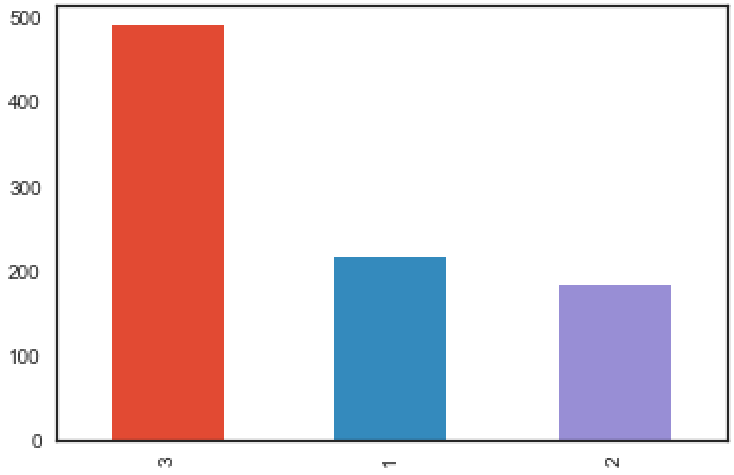
```
plt.figure(figsize=(7,4))
sns.heatmap(df_train.corr(),annot=True,cmap='cubehelix_r') # 상관관계를 Heatmap를 통해 표시합니다.
plt.show()
```



2-11 Bar Plot

In [60]:

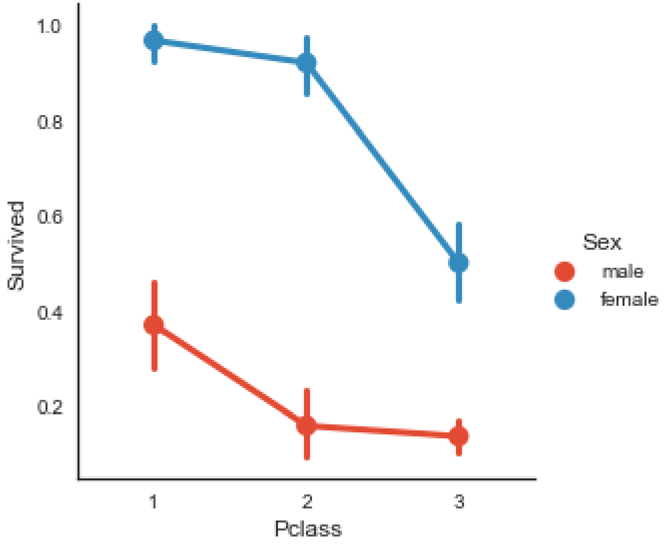
```
df_train['Pclass'].value_counts().plot(kind="bar");
```



2-12 Factorplot

In [61]:

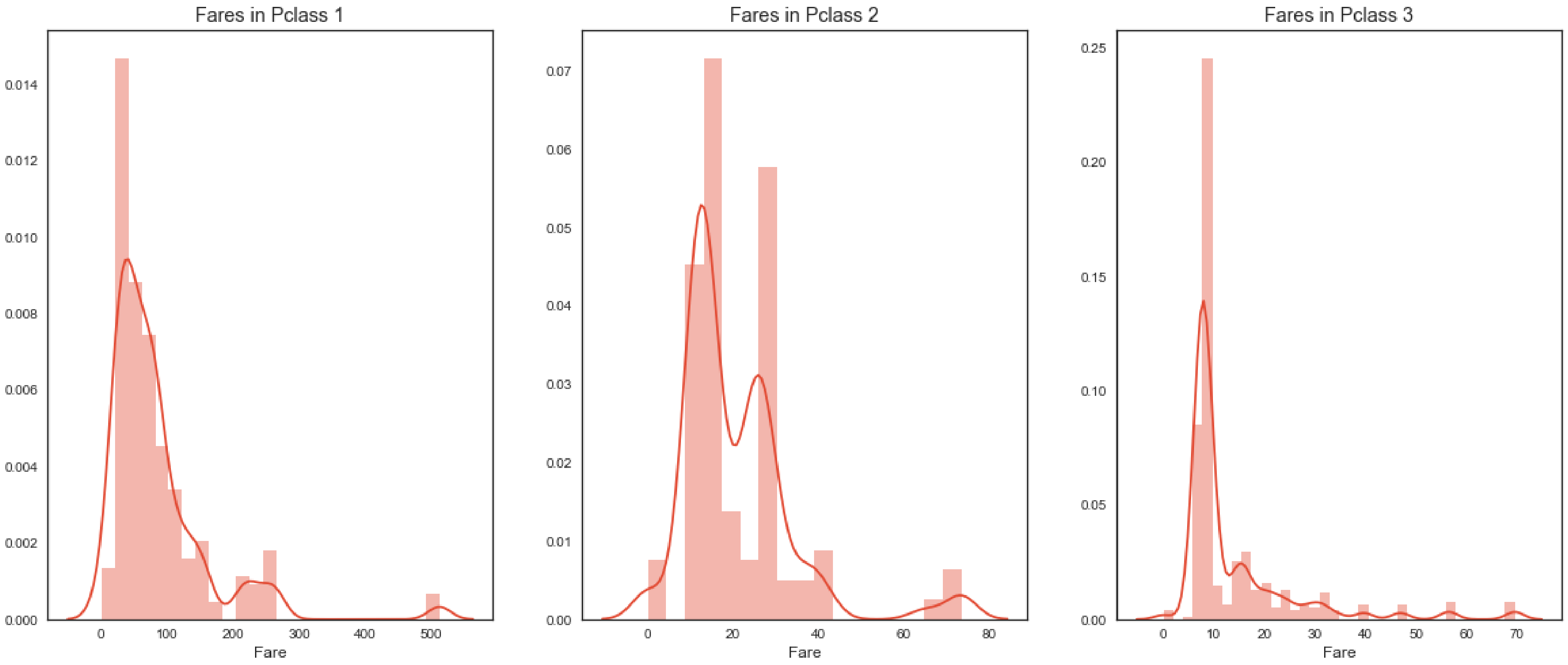
```
sns.factorplot('Pclass','Survived',hue='Sex',data=df_train)
plt.show()
```



2-13 distplot

In [63]:

```
f,ax=plt.subplots(1,3,figsize=(20,8))
sns.distplot(df_train[df_train['Pclass']==1].Fare,ax=ax[0])
ax[0].set_title('Fares in Pclass 1')
sns.distplot(df_train[df_train['Pclass']==2].Fare,ax=ax[1])
ax[1].set_title('Fares in Pclass 2')
sns.distplot(df_train[df_train['Pclass']==3].Fare,ax=ax[2])
ax[2].set_title('Fares in Pclass 3')
plt.show()
```



Ref

https://www.kaggle.com/mjbahmani/a-comprehensive-ml-workflow-with-python (https://www.kaggle.com/mjbahmani/a-comprehensive-ml-workflow-with-python)

Type *Markdown* and LaTeX: α^2

In [ ]: