In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
gender_submission=pd.read_csv('gender_submission.csv')
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')
```

In [4]:
```
gender_submission.head()
```

Out[4]:

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1

In [5]:
```
train.head()
```

Out[5]:

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

In [6]:
```
test.head()
```

Out[6]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [7]:
```
print(train.shape)
train.isnull().sum()
```

```
(891, 12)
```

Out[7]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [8]:
```
print(test.shape)
test.isnull().sum()
```

```
(418, 11)
```

Out[8]:
```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

train dataset

```
train['Cabin'].unique()
```

In [9]:
```
train.count()
```

Out[9]:
```
PassengerId    891
Survived       891
Pclass         891
Name           891
Sex            891
Age            714
SibSp          891
Parch          891
Ticket         891
Fare           891
Cabin          204
Embarked       889
dtype: int64
```

```
survived_td = train[train.Survived==1]
survived_td.head()
```

```
survived_td['Age'].unique()
```

```
print('mean = ',survived_td['Age'].mean(),'\nmode = ',survived_td['Age'].mode(),'\nmedian =
',survived_td['Age'].median(),'\nstd = ',survived_td['Age'].std())
```

```
survived_td['Age'].isnull().sum()
```

In [10]:
```
train.columns
```

Out[10]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [11]:
```
train.head(1)
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25	NaN	S

In [12]:
```
x=train.drop('Survived',axis = 'columns')
```

In [13]:
```
x.head()
```

Out[13]:

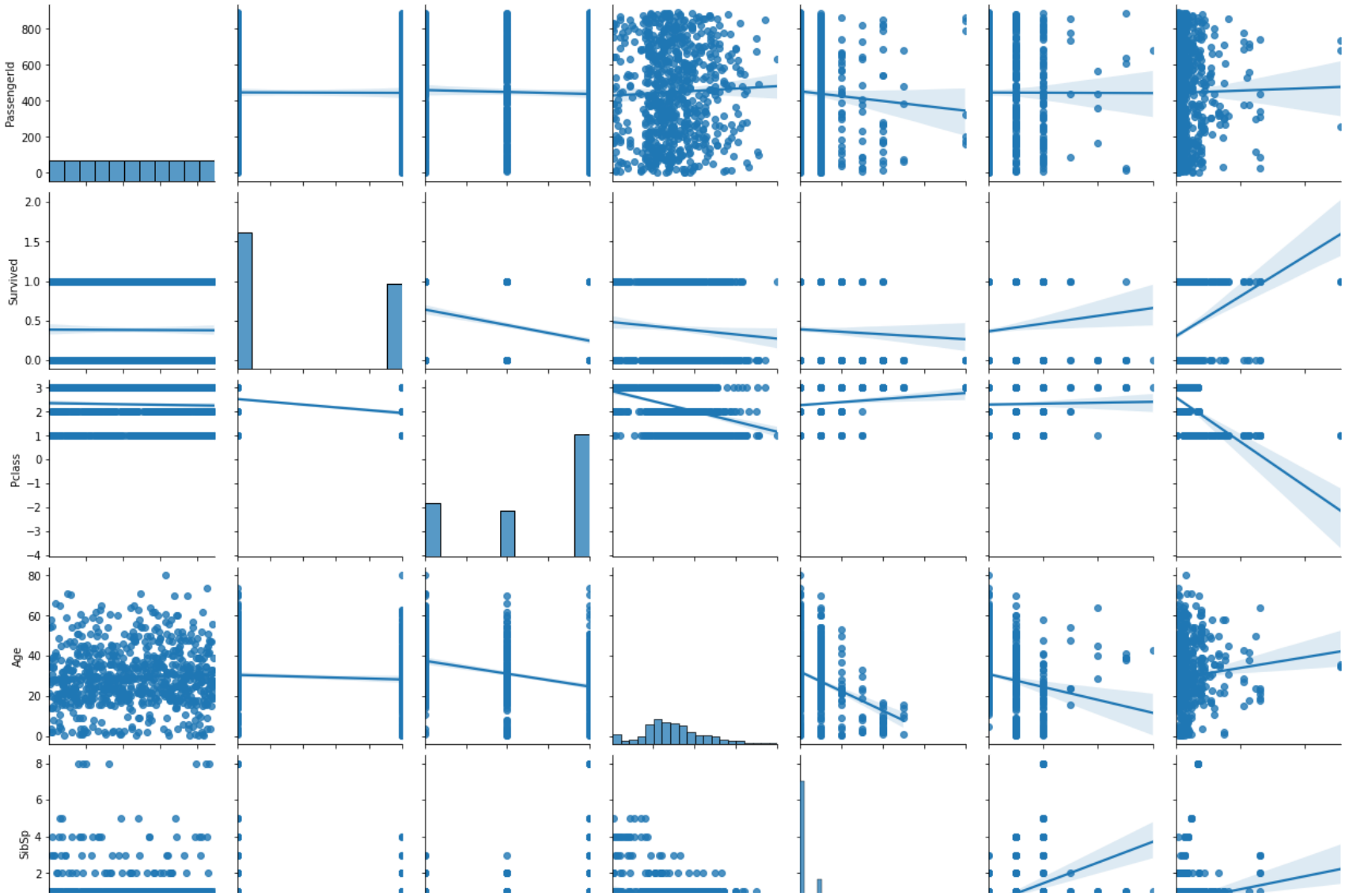
	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250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

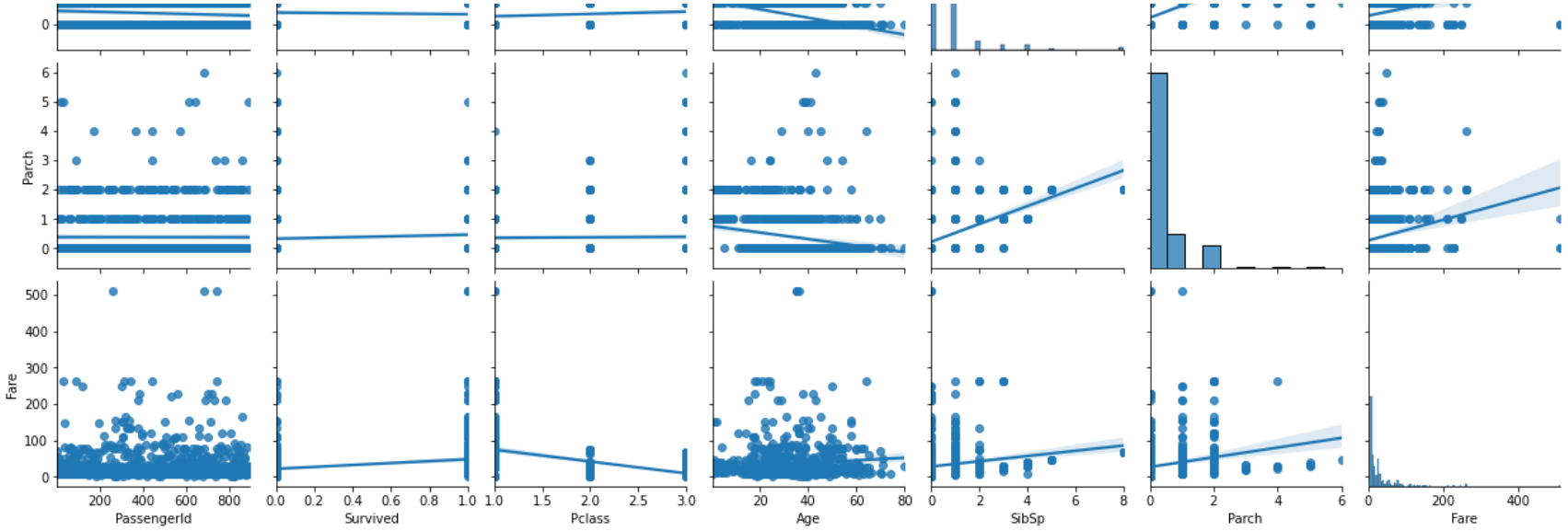
In [14]:
```
x.columns
```

Out[14]:
```
Index(['PassengerId', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch',
       'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [15]:
```
#plt.scatter(x=train['Survived'],y=x['Pclass'])
#plt.scatter(x=train['Survived'],y=x['Age'])
#plt.scatter(x=train['Survived'],y=x['SibSp'])
#plt.scatter(x=train['Survived'],y=x['Parch'])
#plt.scatter(x=train['Survived'],y=x['Fare'])
#plt.scatter(x=train['Survived'],y=x['Embarked'])
sns.pairplot(train,dropna=True,kind='reg')
```

Out[15]: <seaborn.axisgrid.PairGrid at 0x14591772be0>





In [16]:
```
X=x.drop(['PassengerId','Name','Ticket','Cabin'],axis='columns')
X.head()
```

Out[16]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.0	1	0	7.2500	S
1	1	female	38.0	1	0	71.2833	C
2	3	female	26.0	0	0	7.9250	S
3	1	female	35.0	1	0	53.1000	S
4	3	male	35.0	0	0	8.0500	S

In [17]:
```
## checking the values less than 1 so that we can convert back to the normal age
### assusme age in 100 ---> we will multiply age age less than one by  100 for getting age under 100

X[X['Age']<1.0]
```

Out[17]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
78	2	male	0.83	0	2	29.0000	S
305	1	male	0.92	1	2	151.5500	S
469	3	female	0.75	2	1	19.2583	C
644	3	female	0.75	2	1	19.2583	C
755	2	male	0.67	1	1	14.5000	S
803	3	male	0.42	0	1	8.5167	C
831	2	male	0.83	1	1	18.7500	S

In [18]:
```
age = X.Age
age1=[]
for i in age:
    if i<1:
        i=i*100
        age1.append(i)
    else:
        age1.append(i)
age1
age2 = pd.DataFrame(age1)
age2.columns = ['age']
age2.shape
```

Out[18]: (891, 1)

In [19]:
```
x.shape
```

Out[19]: (891, 11)

In [21]:
```
X1 = X.assign(Age = age2)
X1.head()
```

Out[21]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.0	1	0	7.2500	S
1	1	female	38.0	1	0	71.2833	C
2	3	female	26.0	0	0	7.9250	S
3	1	female	35.0	1	0	53.1000	S
4	3	male	35.0	0	0	8.0500	S

.

```
s = train[(train['Survived']==1) & (train['Age'].isnull())].shape
s
```

```
ns = train[(train['Survived']==0) & (train['Age'].isnull())].shape
ns
```

```
train.shape[0] - (train[(train['Survived']==1) & (train['Age'].isnull())].shape[0] +
train[(train['Survived']==0) & (train['Age'].isnull())].shape[0])
```

.

```
X1.Age.std()*2+X1.Age.mean()
```

In [43]:
```
from sklearn.metrics import confusion_matrix
```

In [ ]:
```
```

In [44]:
```
X2 = X1.dropna()
X2.head()
```

Out[44]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.0	1	0	7.2500	S
1	1	female	38.0	1	0	71.2833	C
2	3	female	26.0	0	0	7.9250	S
3	1	female	35.0	1	0	53.1000	S
4	3	male	35.0	0	0	8.0500	S

convert catagories into the num

In [45]:
```
from sklearn.preprocessing import LabelEncoder
```

In [46]:
```
le=LabelEncoder()
```

In [47]:
```
sex=le.fit_transform(X2.Sex)
sex.size
```

Out[47]: 712

In [48]:
```
embarked = le.fit_transform(X2.Embarked)
embarked.size
```

Out[48]: 712

In [49]:
```
X3 = X2.assign(Sex=sex,Embarked=embarked)
X3.head()
```

Out[49]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	2
1	1	0	38.0	1	0	71.2833	0
2	3	0	26.0	0	0	7.9250	2
3	1	0	35.0	1	0	53.1000	2
4	3	1	35.0	0	0	8.0500	2

```
X3['Fare'] = X3['Fare'].round(2)
X3.sort_values(by=['Fare','age']).head()
```

```
X3[X3.Fare==0]
```

```
train[(train.Survived) & (train.Fare==0)]
```

```
train[(train.Fare==0) & (~train.Age.isnull())]
```

```
train.Embarked.unique()
```

```
X4=X3.drop('Age',axis='columns')
X4.head()
```

traing data cleaned properlly (X4 is final dataset)

Test dataset

In [82]:
```
test.isnull().sum()
```

Out[82]:
```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [83]:
```
test.Age.unique()
```

Out[83]:
```
array([34.5 , 47.  , 62.  , 27.  , 22.  , 14.  , 30.  , 26.  , 18.  ,
       21.  ,   nan, 46.  , 23.  , 63.  , 24.  , 35.  , 45.  , 55.  ,
        9.  , 48.  , 50.  , 22.5 , 41.  , 33.  , 18.5 , 25.  , 39.  ,
       60.  , 36.  , 20.  , 28.  , 10.  , 17.  , 32.  , 13.  , 31.  ,
       29.  , 28.5 , 32.5 ,  6.  , 67.  , 49.  ,  2.  , 76.  , 43.  ,
       16.  ,  1.  , 12.  , 42.  , 53.  , 26.5 , 40.  , 61.  , 60.5 ,
        7.  , 15.  , 54.  , 64.  , 37.  , 34.  , 11.5 ,  8.  ,  0.33,
       38.  , 57.  , 40.5 ,  0.92, 19.  , 36.5 ,  0.75,  0.83, 58.  ,
        0.17, 59.  , 14.5 , 44.  ,  5.  , 51.  ,  3.  , 38.5 ])
```

```
[test.Age[test.Age<1]*100]
```

In [84]:
```
x_test=test.drop(['PassengerId','Name','Ticket','Cabin'],axis='columns')
x_test.shape
```

Out[84]:
```
(418, 7)
```

In [85]:
```
x_test.head()
```

Out[85]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	34.5	0	0	7.8292	Q
1	3	female	47.0	1	0	7.0000	S
2	2	male	62.0	0	0	9.6875	Q
3	3	male	27.0	0	0	8.6625	S
4	3	female	22.0	1	1	12.2875	S

In [86]:
```
age_test = x_test.Age
age_test1=[]
for i in age_test:
    if i<1:
        i=i*100
        age_test1.append(i)
    else:
        age_test1.append(i)
age_test1
age_test2 = pd.DataFrame(age_test1)
age_test2.columns = ['age']
age_test2.shape
```

Out[86]: (418, 1)

```
age_test2=age_test2[~(age_test2.age<1)]
age_test2
```

In [87]:
```
X_test=x_test.assign(Age=age_test2)
X_test.head()
```

Out[87]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	34.5	0	0	7.8292	Q
1	3	female	47.0	1	0	7.0000	S
2	2	male	62.0	0	0	9.6875	Q
3	3	male	27.0	0	0	8.6625	S
4	3	female	22.0	1	1	12.2875	S

In [88]:
```
X_test.isnull().sum()
```

Out[88]:
```
Pclass       0
Sex          0
Age         86
SibSp        0
Parch        0
Fare         1
Embarked     0
dtype: int64
```

In [101]:
```
X_test.Age.unique()
```

Out[101]:
```
array([34.5, 47. , 62. , 27. , 22. , 14. , 30. , 26. , 18. , 21. ,  nan,
       46. , 23. , 63. , 24. , 35. , 45. , 55. ,  9. , 48. , 50. , 22.5,
       41. , 33. , 18.5, 25. , 39. , 60. , 36. , 20. , 28. , 10. , 17. ,
       32. , 13. , 31. , 29. , 28.5, 32.5,  6. , 67. , 49. ,  2. , 76. ,
       43. , 16. ,  1. , 12. , 42. , 53. , 26.5, 40. , 61. , 60.5,  7. ,
       15. , 54. , 64. , 37. , 34. , 11.5,  8. , 38. , 57. , 40.5, 92. ,
       19. , 36.5, 75. , 83. , 58. , 59. , 14.5, 44. ,  5. , 51. ,  3. ,
       38.5])
```

In [103]:
```
X_test.Age.size
```

Out[103]:
```
418
```

In [107]:
```
mode = X_test.Age.mode().to_list()
mean = X_test.Age.mean()
std = X_test.Age.std()
rand = std*2+mean
```

In [173]:
```
X_test.Fare.mode()
```

Out[173]:
```
0    7.75
dtype: float64
```

In [175]:
```
X_test.Fare.value_counts()
```

Out[175]:
```
7.7500      21
26.0000     19
8.0500      17
13.0000     17
7.8958      11
            ..
31.6833      1
16.0000      1
53.1000      1
146.5208     1
20.2500      1
Name: Fare, Length: 169, dtype: int64
```

In [ ]:
```

```

In [158]:
```
X_test.Age.value_counts().head(11)
```

Out[158]:
```
24.0    17
21.0    17
22.0    16
30.0    15
18.0    13
27.0    12
26.0    12
23.0    11
25.0    11
29.0    10
45.0     9
Name: Age, dtype: int64
```

In [164]:
```
X_test.Age.fillna(value=np.random.randint(24,30),inplace=True)
```

In [176]:
```
X_test.Fare.fillna(value = 7.7500,inplace=True)
```

In [178]:
```
X_test.isnull().sum()
```

Out[178]:
```
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    0
dtype: int64
```

In [179]:
```
sex_test = le.fit_transform(X_test.Sex)
sex_test.shape
```

Out[179]:
```
(418,)
```

In [180]:
```
embarked_test = le.fit_transform(X_test.Embarked)
embarked_test.shape
```

Out[180]:
```
(418,)
```

In [181]:
```
X_test1 = X_test.assign(Sex = sex_test,Embarked = embarked_test)
X_test1.head()
```

Out[181]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	34.5	0	0	7.8292	1
1	3	0	47.0	1	0	7.0000	2
2	2	1	62.0	0	0	9.6875	1
3	3	1	27.0	0	0	8.6625	2
4	3	0	22.0	1	1	12.2875	2

test dataset is cleaned properlly and (X_test1) final result

Ready for data cleaning

In [184]:
```
print('shape = ',X_test1.shape)
X_test1.head()
```

```
shape =  (418, 7)
```

Out[184]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	34.5	0	0	7.8292	1
1	3	0	47.0	1	0	7.0000	2
2	2	1	62.0	0	0	9.6875	1
3	3	1	27.0	0	0	8.6625	2
4	3	0	22.0	1	1	12.2875	2

In [186]:
```
print('shape = ',X3.shape)
X3.head()
```

```
shape =  (712, 7)
```

Out[186]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	2
1	1	0	38.0	1	0	71.2833	0
2	3	0	26.0	0	0	7.9250	2
3	1	0	35.0	1	0	53.1000	2
4	3	1	35.0	0	0	8.0500	2

for test dataset for dv variables

In [187]:
```
y_test=gender_submission
y_test.head()
```

Out[187]:

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1

In [188]:
```
y_test0=y_test.drop('PassengerId',axis='columns')

y_test0.head()
```

Out[188]:

	Survived
0	0
1	1
2	0
3	0
4	1

train dataset for idv

In [189]:
```
y_train = train.drop(['PassengerId','Name','Ticket','Cabin'],axis= 'columns')
y_train.isnull().sum()
```

Out[189]:
```
Survived      0
Pclass        0
Sex           0
Age         177
SibSp         0
Parch         0
Fare          0
Embarked      2
dtype: int64
```

In [190]:
```
dv = y_train.dropna()
dv = pd.DataFrame(dv.Survived)
```

In [191]:
```
print(X3.shape)
dv.shape
```

```
(712, 7)
```

Out[191]: (712, 1)

In [192]:
```
dv.head()
```

Out[192]:

	Survived
0	0
1	1
2	1
3	1
4	0

In [193]:
```
dv_train = dv
idv_train = X3
dv_test = y_test0
idv_test = X_test1
```

In [194]:
```
print('dv_train = ',dv_train.shape,', idv_train = ',idv_train.shape,'\ndv_test = ',dv_test.shape,', idv_test = '
```

```
dv_train =  (712, 1) , idv_train =  (712, 7)
dv_test =  (418, 1) , idv_test =  (418, 7)
```

In [195]:
```
idv_test.tail()
```

Out[195]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
413	3	1	29.0	0	0	8.0500	2
414	1	0	39.0	0	0	108.9000	0
415	3	1	38.5	0	0	7.2500	2
416	3	1	29.0	0	0	8.0500	2
417	3	1	29.0	1	1	22.3583	0

```
train(11) --> { except survived }  ==>
test(1) ---> { survived }
```

training the data using the machine learning

DecisiontreeClassifier, RandomForestreClassifier, SVM, LogisticRegression

In [196]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
```

```
rf_model=RandomForestClassifier(n_estimators=1000,max_features=2,oob_score=True)
features=idv_train.columns
rf_model.fit(idv_train,dv_train)
print('oob accuracy = ',rf_model.oob_score_)
```

```
for features,imp in zip(features,rf_model.feature_importances_):
    print(features,imp)
```

In [247]:
```
lr = LogisticRegression(solver='newton-cg')
dt = DecisionTreeClassifier()
rf = RandomForestClassifier(n_estimators=1000,max_features=5,oob_score=True)
svm = SVC(kernel='linear',gamma='auto')
```

In [248]:
```
models = [lr,dt,rf,svm]
model_name = ['logistic','decision','random forest','svm']
for i in range(len(models)):

    #models[i].fit(idv_train,dv_train)
    print(model_name[i],'score =',models[i].fit(idv_train,dv_train).score(idv_train,dv_train))
```

```
logistic score = 0.7851123595505618
decision score = 0.9859550561797753
random forest score = 0.9859550561797753
svm score = 0.7794943820224719
```

In [249]:
```
lr_pred = lr.predict(idv_test)
lr_pred.size
```

Out[249]:
```
418
```

In [250]:
```
pred_model = []
for i in range(len(models)):

    print(model_name[i],'score =',models[i].fit(idv_train,dv_train).score(idv_train,dv_train))
    pred_model.append(pd.DataFrame([models[i].fit(idv_train,dv_train).predict(idv_test)]))
```

```
logistic score = 0.7851123595505618
decision score = 0.9859550561797753
random forest score = 0.9859550561797753
svm score = 0.7794943820224719
```

In [255]:
```
pred_modl = pd.concat([pred_model[0],pred_model[1],pred_model[2],pred_model[3]])
```

In [256]:
```
pred_modl = pred_modl.T
```

In [257]:
```
pred_modl.head(3)
```

Out[257]:

	0	0	0	0
0	0	0	0	0
1	0	0	0	1
2	0	1	1	0

In [258]:
```
pred_modl.columns = model_name
print(pred_modl.shape)
pred_modl.head()
```

```
(418, 4)
```

Out[258]:

	logistic	decision	random forest	svm
0	0	0	0	0
1	0	0	0	1
2	0	1	1	0
3	0	1	1	0
4	1	0	0	1

```
pred_modl.reset_index(drop=True,inplace=True)
gender_submission.reset_index(drop=True,inplace=True)
```

In [259]:
```
check = pd.concat([gender_submission,pred_modl],axis='columns')
check.head(10)
```

Out[259]:

	PassengerId	Survived	logistic	decision	random forest	svm
0	892	0	0	0	0	0
1	893	1	0	0	0	1
2	894	0	0	1	1	0
3	895	0	0	1	1	0
4	896	1	1	0	0	1
5	897	0	0	0	0	0
6	898	1	1	0	0	1
7	899	0	0	0	0	0
8	900	1	1	0	1	1
9	901	0	0	0	0	0

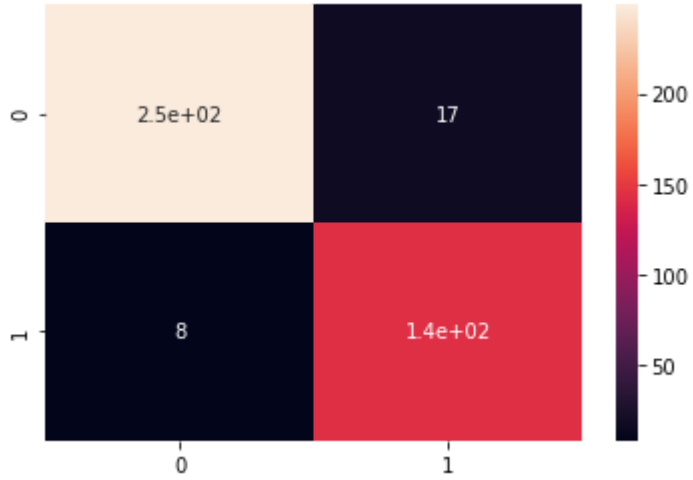
checking the how much it predicted properly using confusion matrix

In [260]:
```
from sklearn.metrics import confusion_matrix
```

In [270]:
```
lr_c_matrix = confusion_matrix(check.Survived,check.logistic)
print(lr_c_matrix)
sns.heatmap(lr_c_matrix, annot=True)
```

```
[[249  17]
 [  8 144]]
```

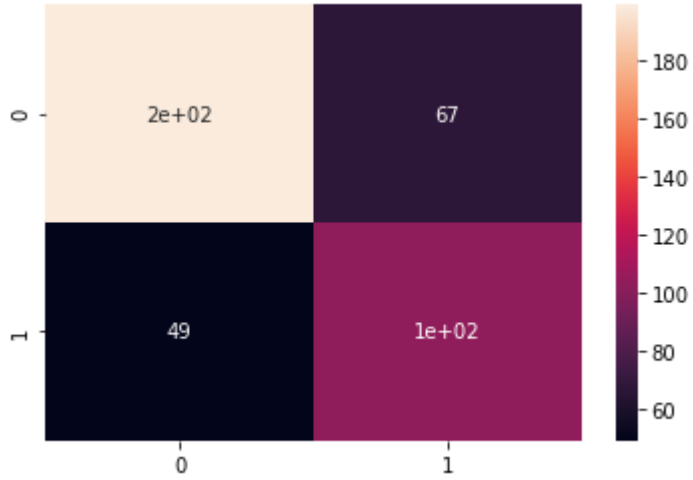
Out[270]: <AxesSubplot:>



In [271]:
```
dt_c_matrix = confusion_matrix(check.Survived,check.decision)
print(dt_c_matrix)
sns.heatmap(dt_c_matrix, annot=True)
```

```
[[199  67]
 [ 49 103]]
```

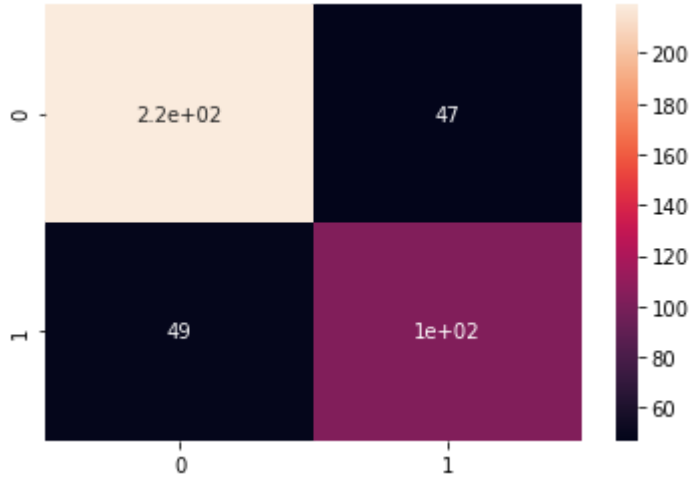
Out[271]: <AxesSubplot:>



In [272]:
```
rf_c_matrix = confusion_matrix(check.Survived,check['random forest'])
print(rf_c_matrix)
sns.heatmap(rf_c_matrix, annot=True)
```

```
[[219  47]
 [ 49 103]]
```

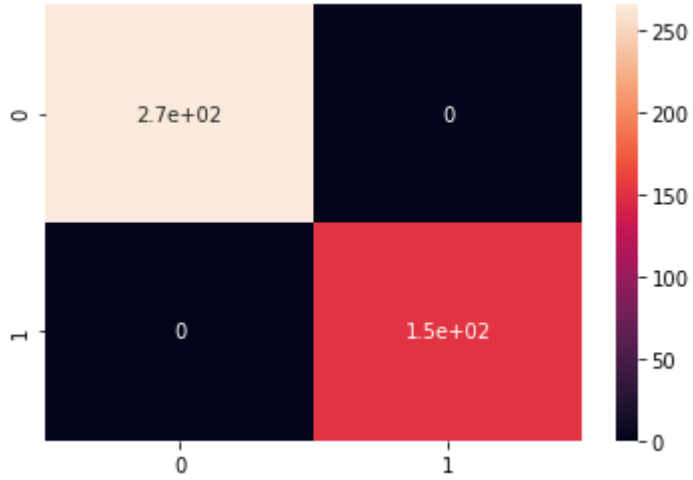
Out[272]: <AxesSubplot:>



In [273]:
```
svm_c_matrix = confusion_matrix(check.Survived,check.svm)
print(svm_c_matrix)
sns.heatmap(svm_c_matrix, annot=True)
```

```
[[266   0]
 [  0 152]]
```

Out[273]: <AxesSubplot:>



SVM worked properly and predicted accurately ¶

parameter used in svm is { kernal = 'linear', gamma = 'auto'}