Surviving Titanic

Introduction and initial question

This time we will take a look at a data set containing passenger data from the Titanic and see if we can find any patterns in the data pointing to if sex, age, or social class had influence over the odds of surviving the disastrous shipwreck.

Was it possibly so the the upper class were favored and had a higher survival rate than other passengers? Perhaps age made a difference? And was it really ladies first even for the lifeboats on the sinking ship?

Below we will take a look at and analyze the passenger data to see if we can find the answers to these questions. However, before we start let's import the libraries we need for our analysis.

In [1]:
```
# Importing numpy and pandas for handy data handling.
import numpy as np
import pandas as pd

# Matplotlib for plotting our data and seaborn's styling for eyecandy.
import matplotlib.pyplot as plt
import seaborn

# We will use statsmodels logistic regression model last to analyze
# the different probabilities of survival depending on age, gender, and class.
import statsmodels.api as sm

# Inline plots.
%matplotlib inline
```

Now we are ready to tackle our choosen data set.

The dataset

The data set itself is taken from the following Kaggle competition (https://www.kaggle.com/c/titanic/data) which works as an introduction to machine learning. Today we will use the same data but focus more on our analysis of the data than making predictions.

We start by reading in the data using the pandas CSV reader.

In [2]:
```
passengers = pd.read_csv('titanic-data.csv')
passengers.head(3)
```

Out[2]:

	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
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282

It seems like the data has been read with almost no problem at all, but there are some columns in the given set that needs further explanation. The following explanations are given on the Kaggle competition page.

Variable	Definition	Key
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

We can see that there are some 'NaN' values present in the cabin column so just to be sure we will take an extra look at our data with the `.info()` method.

In [3]:
```
# Check info about the data we read with .read_csv()
passengers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

We can see that in addition to the Cabin, Age, and Embarked columns are also missing values. We either need to estimate the missing values or drop them. We could estimate missing ages with either median or average values, but for our analysis the non-null 714 values is enough. We choose to just drop the rows with null values.

Cabin has to many missing values and according to wikipedia also a possible bias towards first class passengers. However, something that would be interesting to look up is for example if cabin proximity to the few lifeboats that were available influenced the possibility for survival. This is out of scope for this time so instead we choose to just drop the column instead.

Also, upon further inspection it seems like SibSp/Parch and ticket numbers don't have any much connection making it hard to make sense of which passenger are related to which. There might be some interesting analysis in answering the question if being in a group increased or actually decreased the potential for survival. But, once again this might be a bit to much work for this time so we choose to drop these three columns instead.

The same reasoning goes for the Name, PassengerId, and Fare columns which we also choose to drop.

In [4]:
```
# Dropping rows with NaN values
passengers.dropna(subset=['Age'], inplace=True)

# Dropping unneeded columns
passengers.drop('Cabin', axis=1, inplace=True)
passengers.drop('Embarked', axis=1, inplace=True)

passengers.drop('SibSp', axis=1, inplace=True)
passengers.drop('Parch', axis=1, inplace=True)
passengers.drop('Ticket', axis=1, inplace=True)

passengers.drop('Name', axis=1, inplace=True)
passengers.drop('PassengerId', axis=1, inplace=True)
passengers.drop('Fare', axis=1, inplace=True)
```

Lastly, to make column names more coherent we rename the Pclass column to just Class and to better describe to contents of each column we convert the Sex, Class, and Survived column's data types from Strings to Categories, and Integers to booleans respectively.

In [5]:
```
passengers = passengers.rename(columns={'Pclass': 'Class'})

passengers['Sex'] = passengers.Sex.astype('category')
passengers['Class'] = passengers.Class.astype('category')

passengers['Survived'] = passengers.Survived.apply(bool)
```

This gives us the below data frame to work with.

In [6]:
```
passengers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 714 entries, 0 to 890
Data columns (total 4 columns):
Survived    714 non-null bool
Class       714 non-null category
Sex         714 non-null category
Age         714 non-null float64
dtypes: bool(1), category(2), float64(1)
memory usage: 13.3 KB
```

Descriptive stats

```
In [7]: passengers['Age'].describe()
```

```
Out[7]: 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
```

```
In [8]: passengers['Sex'].describe()
```

```
Out[8]: count      714
        unique       2
        top       male
        freq       453
        Name: Sex, dtype: object
```

```
In [9]: passengers['Class'].describe()
```

```
Out[9]: count     714
        unique      3
        top         3
        freq      355
        Name: Class, dtype: int64
```

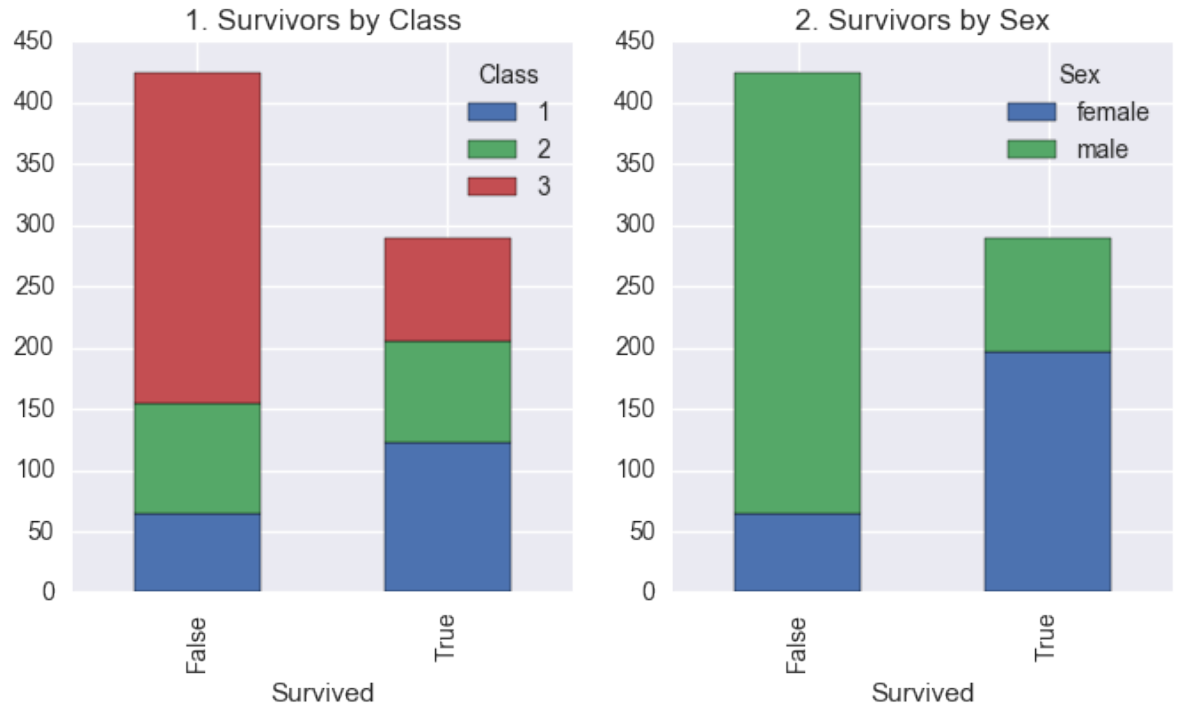
```
In [10]: passengers['Survived'].describe()
```

```
Out[10]: count       714
         unique        2
         top       False
         freq        424
         Name: Survived, dtype: object
```

Based on the above, we can see here that according to our sample the typical passenger was a male around 30 travelling in 3 class and, unfortunately, a casualty from the shipwreck. Worth noting is that after we cleaned up the data we have no missing values in any of our remaining columns.

Lets try to plot some different relationships between class, sex, and age for the survivors and non-survivors in our data set to see if we can find something interesting. We start out with plotting comparisons for the composition of passenger class and sex between survivors and casualties.

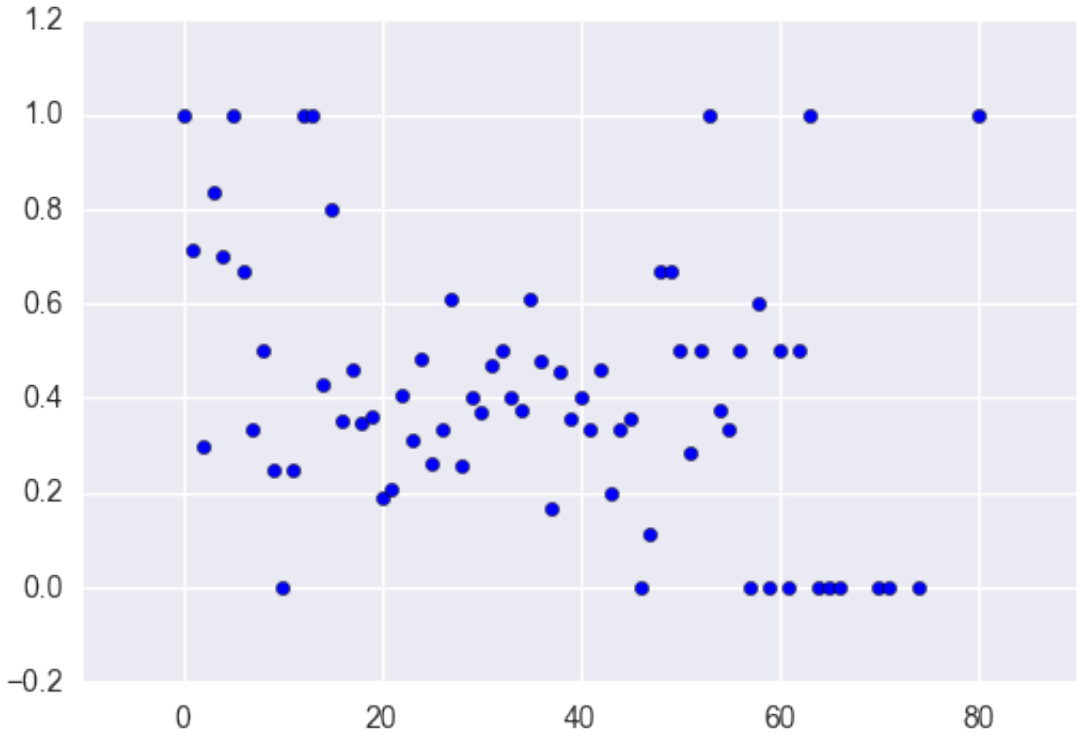
In [11]:
```
# Create a grid with two columns for display
fig, axs = plt.subplots(ncols=2, figsize=(8, 4))

# Group passengers by Survived and Class then plot the results in a
bar graph
survived_by_class = passengers.groupby(['Survived','Class']).size()
.unstack()
survived_by_class.plot(kind='bar', stacked=True, ax=axs[0])
axs[0].set_title('1. Survivors by Class')

# Do the same as above but with Survived and Sex
survived_by_sex = passengers.groupby(['Survived', 'Sex']).size().un
stack()
survived_by_sex.plot(kind='bar', stacked=True, ax=axs[1])
axs[1].set_title('2. Survivors by Sex')

# Display plots
plt.show()
```



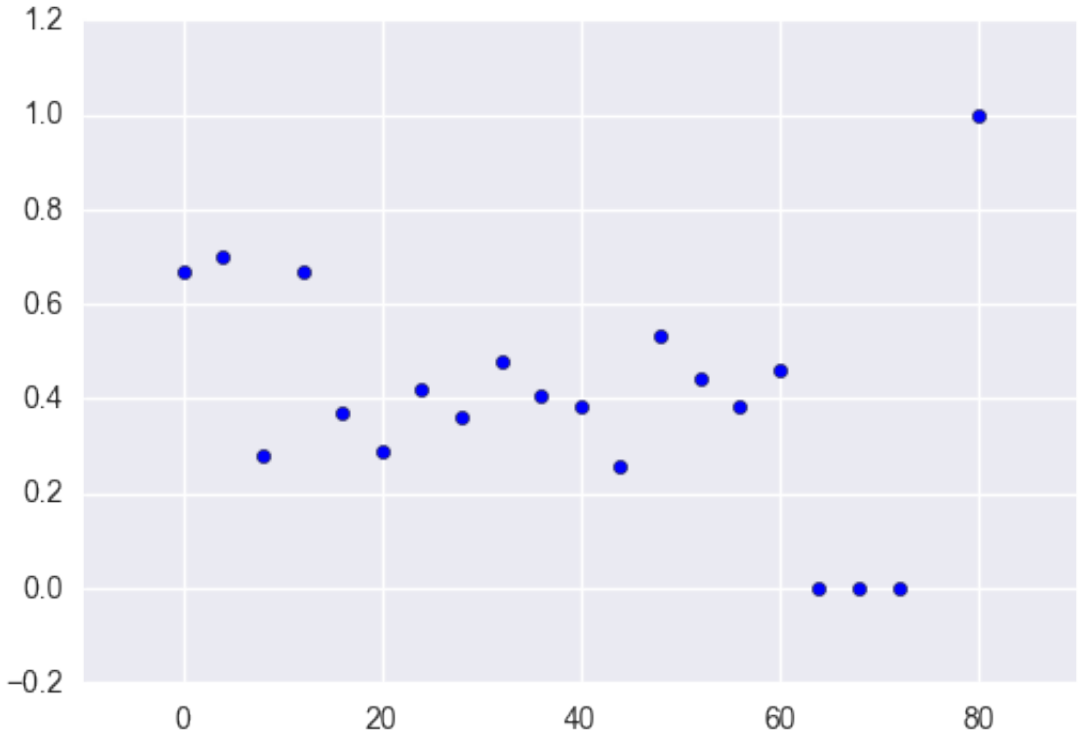
From the above bar graphs we can see that there is a big change in ratio between male and female, and between 3 class 1/2 class passengers represented in the two different categories (Survived: False/True). It do look as if women had a higher chance of survival than men and 3 class passengers a much lower chance than 1 class passengers. To really make sure that this isn't just in our sample we will perform tests to see if the two deviations are statistically significant or not.

We also tried plotting the survival rate for passengers born the same year but too much noise did make it hard to get anything out of the plot. By combining the ages in to intervals of 4 we can keep some of the noise down. Judning from the adjusted graph it looks like we might have a correlation between higher age and an increased mortality rate. This will also be a question we would like to answer.

In [12]:
```
#Grouped by age in full years
survival_by_age = passengers.groupby(lambda x: int(passengers.loc[x
].Age)).Survived
survival_rate_by_age = survival_by_age.apply(lambda x: x[x == True]
.count()/(x.count() * 1.0))
plt.scatter(survival_rate_by_age.index, survival_rate_by_age);
```



In [13]:
```
#Group by age in intervals of 4
survival_by_age_group = passengers.groupby(lambda x: int(passengers
.loc[x].Age/4)*4).Survived
survival_rate_by_age_group = survival_by_age_group.apply(lambda x:
x[x == True].count()/(x.count() * 1.0))
plt.scatter(survival_rate_by_age_group.index, survival_rate_by_age_
group);
```



Chi squared test for independence

After looking at the above three variables and plots we can hypothesize about there being three factors that all had effect on the chance of survival.

1. Class, lower passenger class also meant lower probability for survival.
2. Sex, women had a higher probability for survival than men.
3. Age, higher age also meant a decreased chance of survival.

We will start by using a chi squared test for independence (https://en.wikipedia.org/wiki/Chi-squared_test) to test if the frequency of survival for our passengers and dependent on the first two factors, Class and Gender, frequencies.

The test is possible to perform since all our involved variables are categorical, meaning that all our outcomes are mutually exclusive (you cannot be dead and alive, or in first and second class at the same time) and the total probability for all outcomes adds up to 1 (meaning you have to be either be dead or alive etc).

Explained very shortly to do a chi squared test we calculate our chi square statistic, χ^2 , as a normalized sum of the squared deviations between observed and theoretical frequencies and then compare it to a critical value (found in a table) based on our confidence level and degrees of freedom. Expressed mathematically it looks something like the below.

$$\chi^2 = \sum_n \frac{(O_i - E_i)^2}{E_i}$$

where

χ^2 = the chi statistic
O_i = observations of type i
$E_i = Np_i$ = expected frequency for type i
N = Total number of observations
n = number of types of observations

To test for variable indepence we formulate the following hypothesises.

H_0: The variables are independent.
H_a: The variables are **not** independent.

To start our calculations we first define a function for calculating the χ^2 -statistic.

Then we define how to perform the chi square test.

In [14]:
```
def chi2_test(data, var1, var2):
    '''
    Chi^2 test that checks a defined variable(var2) against another (var1) boolean variable for dependency.
    Returns the chi statistic for the defined variable(var2).
    '''
    var1_by_var2 = pd.DataFrame(data.groupby([var2, var1]).size().unstack()[True])
    rate = data[data[var1] == True][var1].count() / (data[var1].count()*1.0)

    var1_by_var2['Total'] = data.groupby([var2]).size()
    var1_by_var2['Expected'] = var1_by_var2['Total'] * rate
    var1_by_var2['Difference'] = var1_by_var2[True] - var1_by_var2['Expected']
    var1_by_var2['Difference2'] = var1_by_var2['Difference']**2
    var1_by_var2['Proportion'] = var1_by_var2['Difference2'] / var1_by_var2['Expected']

    return var1_by_var2['Proportion'].sum()
```

Degrees of freedom is defined as $(rows - 1)(columns - 1)$. In our case with 2 possible outcomes for survival and 3 for class we get $df = 1 * 2 = 2$.
Further, our α(confidence level) for the test is 0.05 which gives us a critical value of 5.99 from our χ^2 table (https://en.wikipedia.org/wiki/Chi-squared_distribution#Table_of_.CF.872_values_vs_p-values).

In [15]:
```
chi2_test(passengers, 'Survived', 'Class')
```

Out[15]:
```
55.168348596145215
```

The statistic value is higher than our critical value which means we reject our H_0 that the two variables survival and class are independent with a 95% probability.

Similarly for survival and sex we get $df = 1 * 1 = 1$ with two outcomes for both survival and sex. Using the same α gives us a χ^2 critical value of 3.84.

In [16]:
```
chi2_test(passengers, 'Survived', 'Sex')
```

Out[16]:
```
123.1012003475312
```

Once again we get a statistic value larger than our critical value meaning that we once again reject the H_0. This means there are a dependency between sex and survival rate with a probability of 95%.

Logistical regression for probabilities

Next we will take our analysis one step further and investigate the relation between Age and survival with help from a logistical regression model from the statsmodels package. For another more extensive example have a look at this blog post (http://blog.yhat.com/posts/logistic-regression-python-rodeo.html) which I also used as a reference for the following analysis.

Since we already have verified that both class and sex also influences survival we choose to add these variables as well to the regression model. In order to add these two non-continuous variables we need to create some additional dummy variables to represent the different values for class and age as simple True(1) or False(0) variables. This is simple to do using pandas .get_dummies() method for our data frame.

In [17]:
```
# Add dummy variables for class and sex.
passenger_dummies = passengers.join(
    pd.get_dummies(passengers['Class'], prefix='Class'))
passenger_dummies = passenger_dummies.join(
    pd.get_dummies(passengers['Sex']))

# We drop the Class and Sex columns from the table since they aren'
t needed.
passenger_dummies.drop('Class', axis=1, inplace=True)
passenger_dummies.drop('Sex', axis=1, inplace=True)

# To create a baseline, our typical passenger, we drop the class 3
and male column.
passenger_dummies.drop('Class_3', axis=1, inplace=True)
passenger_dummies.drop('male', axis=1, inplace=True)

passenger_dummies.head()
```

Out[17]:

	Survived	Age	Class_1	Class_2	female
0	False	22.0	0	0	0
1	True	38.0	1	0	1
2	True	26.0	0	0	1
3	True	35.0	1	0	1
4	False	35.0	0	0	0

Next step is to feed our data into the logistic regression model from the statsmodel package and fit it to the inputted data.

In [18]:
```
logit = sm.Logit(passenger_dummies['Survived'],
                 passenger_dummies[passenger_dummies.columns[1:]])

# Fit the model
result = logit.fit()

result.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.474695
         Iterations 6
```

Out[18]:

Logit Regression Results

Dep. Variable:	Survived	**No. Observations:**	714
Model:	Logit	**Df Residuals:**	710
Method:	MLE	**Df Model:**	3
Date:	Fri, 28 Apr 2017	**Pseudo R-squ.:**	0.2972
Time:	08:48:17	**Log-Likelihood:**	-338.93
converged:	True	**LL-Null:**	-482.26
		LLR p-value:	7.700e-62

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Age	-0.0682	0.006	-12.145	0.000	-0.079	-0.057
Class_1	2.6026	0.283	9.196	0.000	2.048	3.157
Class_2	0.9867	0.233	4.232	0.000	0.530	1.444
female	2.1893	0.198	11.035	0.000	1.800	2.578

Looking at the coefficients for Age, Class_1, Class_2, and female we can see that we have a negative relation between and survival while class and sex have positive relations, all consistent with what we saw in our plots in the beginning. It is also important to note that the confidence interval for all coefficients are strictly negative or positive meaning there are no question that each of the coefficients have influence over our survival rate.

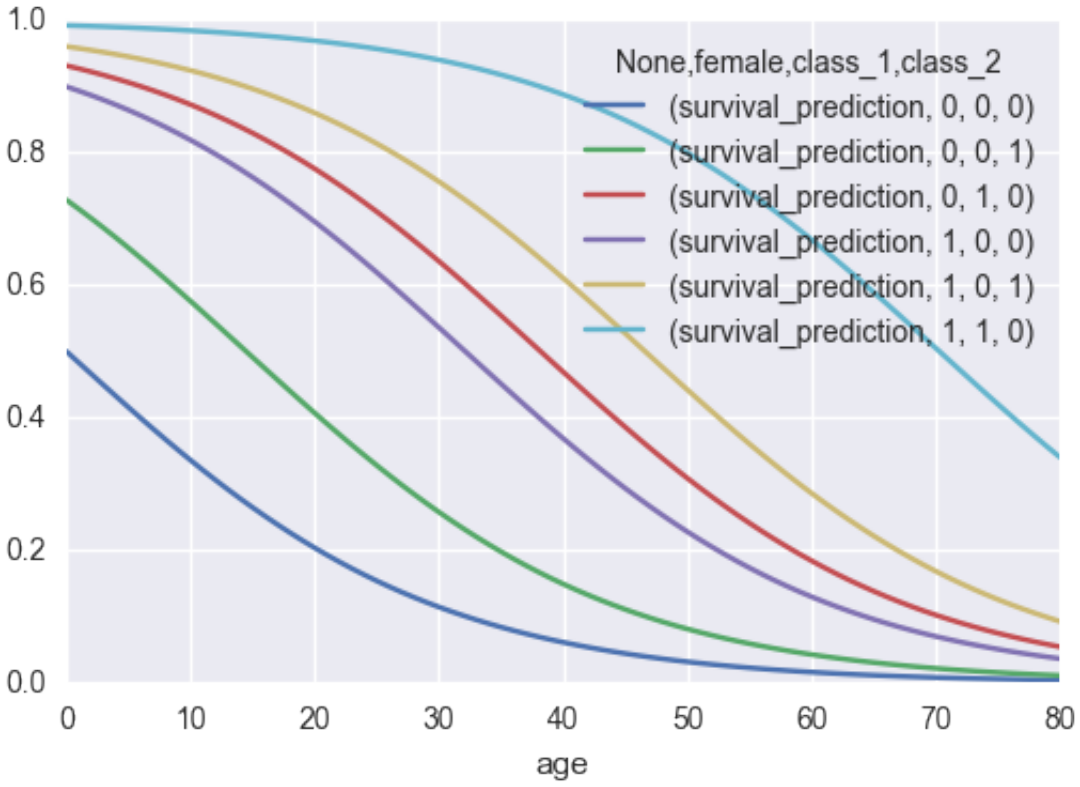
As a last trick we plot all the different possibilities for survival based on age for each combination of our categorical variables so that we can easier see what passengers had the highest and lowest possiblity of survival during the shipwreck.

In [19]:
```
# Last step, visualizing the probability for all possible data poin
ts
ages = np.linspace(0, 80, 81, dtype=int)

# Define cartesian function
def cartesian(arrays, out=None):
    arrays = [np.asarray(x) for x in arrays]
    dtype = arrays[0].dtype

    n = np.prod([x.size for x in arrays])
    if out is None:
        out = np.zeros([n, len(arrays)], dtype=dtype)

    m = n / arrays[0].size
    out[:,0] = np.repeat(arrays[0], m)
    if arrays[1:]:
        cartesian(arrays[1:], out=out[0:m,1:])
        for j in xrange(1, arrays[0].size):
            out[j*m:(j+1)*m,1:] = out[0:m,1:]
    return out

# Enumerate all possible combinations of dummy variables (0=male, 1
=female)
combinations = pd.DataFrame(cartesian([ages, [1, 2, 3], [0, 1]]))

# Recreate dummies
combinations.columns = ['age', 'class', 'female']

dummy_classes = pd.get_dummies(combinations['class'])
dummy_classes.columns = ['class_1', 'class_2', 'class_3']

dummy_sexes = pd.get_dummies(combinations['female'])
dummy_sexes.columns = ['male', 'female']

# Keep what we need for predictions and join with dummies

combinations = pd.DataFrame(combinations['age']).join(dummy_classes
.ix[:, :'class_2'])
combinations = combinations.join(dummy_sexes.ix[:, 'female':])

combinations['survival_prediction'] = result.predict(combinations)

# Predicted survival probabilities for a average age passenger.
combinations[180:186]
```

Out[19]:

	age	class_1	class_2	female	survival_prediction
180	30	1	0	0	0.635962
181	30	1	0	1	0.939755
182	30	0	1	0	0.257695
183	30	0	1	1	0.756086
184	30	0	0	0	0.114586
185	30	0	0	1	0.536085

In [21]:
```
#Pivoting table for plotting.
pivot = pd.pivot_table(combinations,
                       values=['survival_prediction'],
                       index=['age'],
                       columns=['female', 'class_1', 'class_2'])

# And plot the resulting graph.
pivot.plot();
```



Conclusion

From the last plot we can see that our analysis show a trend towards relation between higher age and lower chance for surviving the shipwreck. Further the different combination of passenger sex and class can be ranked the following way from highest probability to survive compared to other passengers of the same age to lowest. (Color from above plot in brackets)

1. First class women (Turquoise)
2. Second class women (Yellow)
3. First class men (Red)
4. Third class women (Purple)
5. Second class men (Green)
6. Third class men (Blue)

Now I believe we can answer our first initial question. Did age, class (using passenger class as a proxy here), or sex influence the probability of survival? Based on our findings we can with statistical certainty say, yes.

We can also see from our ranked list above that women had a higher probability than men of surviving the shipwreck and that upper class passengers were being favored over other passengers. However, sex did almost always have priority over class seen by the fact that the only male passengers having higher chance of survival than any women in their age group were first class male passengers beating Third class women by a hair. Personally I would at least like to draw the conclusion that the majority of passengers, second and third class men, did indeed behave like gentlemen even on the sinking Titanic.