titanicFate-class

April 21, 2022

[78]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
#import matplotlib as plt
import seaborn as sns

df = pd.read_csv("C:
 ↪\\Users\\guibs\\Documents\\GitHub\\SGD\\Labs\\lab8_class\\data\\titanic.csv")
df
```

[78]:
```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           …       …     …
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                 Name     Sex   Age  SibSp  \
0                             Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                                Heikkinen, Miss. Laina  female  26.0      0
3          Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                              Allen, Mr. William Henry    male  35.0      0
..                                                   …     …   …    …
886                              Montvila, Rev. Juozas    male  27.0      0
887                       Graham, Miss. Margaret Edith  female  19.0      0
888           Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                              Behr, Mr. Karl Howell    male  26.0      0
890                                Dooley, Mr. Patrick    male  32.0      0

     Parch            Ticket     Fare Cabin Embarked
```

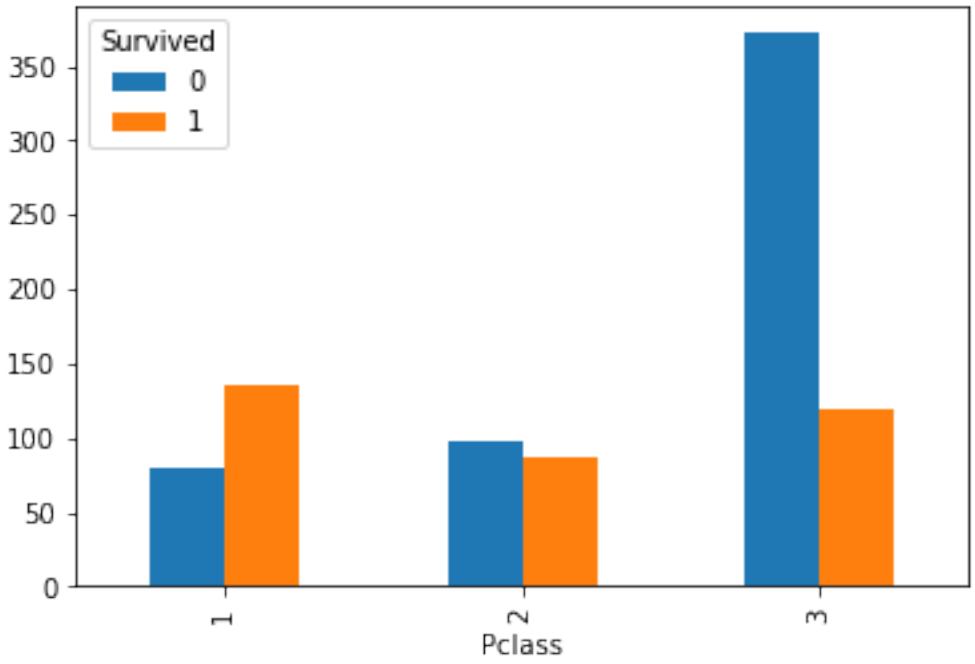
```
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     …               …       …   …      …
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

1 How do you correlate passenger class with survival chances?

[79]:
```
pd.crosstab(df.Survived, df.Pclass, margins=True)
```

[79]:
```
Pclass      1    2    3  All
Survived
0          80   97  372  549
1         136   87  119  342
All       216  184  491  891
```

[80]:
```
temp4 = pd.crosstab(df['Pclass'], df['Survived'])
temp4.plot(kind='bar')
plt.show()
```

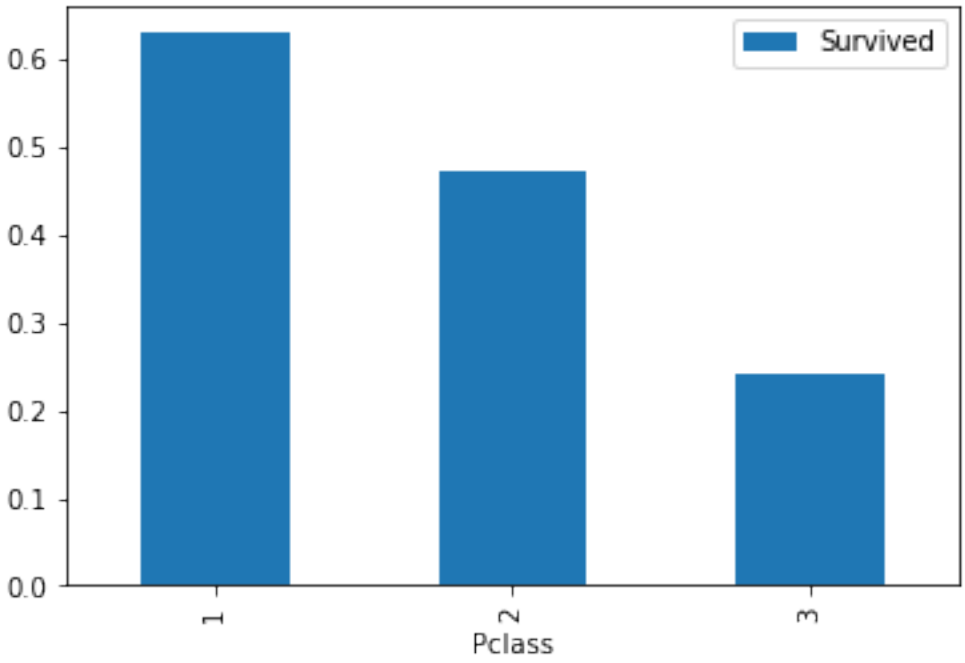


2 What is the percenbtage of survivors in each passenger class, therefore, how did the passenger class influence survivability?

Probability of surviving in each class. First class, 63% survived, Second Class 47% survived, Third class 24% survived.

[81]:
```
# 62% dos class 1 sobreviveram
# 48% dos class 2 sobreviveram
# 25% dos class 3 sobreviveram
temp1 = df['Pclass'].value_counts(ascending=False)

temp2 = df.pivot_table(values='Survived',index=['Pclass'],aggfunc=lambda x: x.
 ↪mean())

print ('Number of passengers in each class:')
print (temp1)

print ('Percentage of survivals in each class:')
print (temp2 )


temp2.plot(kind='bar')
plt.show()
```

```
Number of passengers in each class:
3    491
1    216
2    184
Name: Pclass, dtype: int64
Percentage of survivals in each class:
        Survived
Pclass
1       0.629630
2       0.472826
3       0.242363
```



3 How did the sex influence survival

Females in all classes had a higher change of surviving, because of "Women and children priorities"

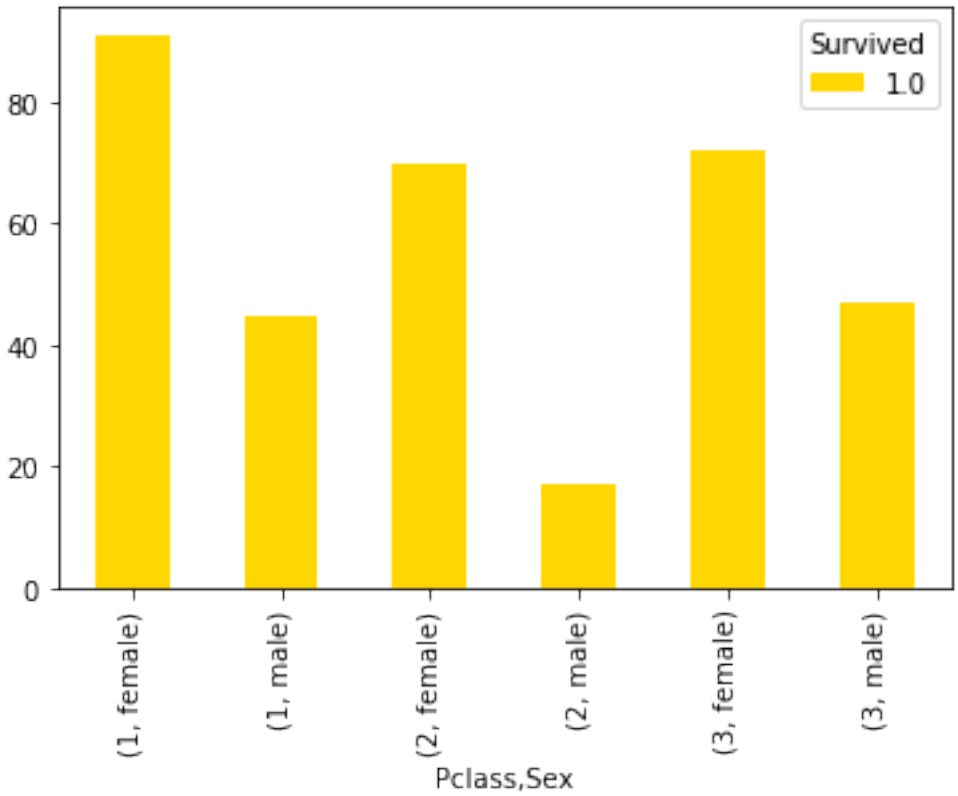
[82]:
```
pd.crosstab(([df['Pclass'],df['Sex']]),df['Survived'])
```

[82]:
```
Survived         0   1
Pclass Sex
1      female    3  91
       male     77  45
2      female    6  70
       male     91  17
3      female   72  72
```

```
male    300  47
```

4 Some simple transformation...

[83]:
```
tempx = pd.crosstab(([df['Pclass'],df['Sex']]),df['Survived'].
 ↪where(df['Survived']==1))

tempx.plot(kind='bar', color=['gold','blue'], grid=False)
plt.show()
```



[84]:
```
def status(feature):
        print ('Processing',feature,': ok' )

def process_family():
        # introducing a new feature : the size of families (including the␣
 ↪passenger)
        df['FamilySize'] = df['Parch'] + df['SibSp'] + 1

        # introducing other features based on the family size
        df['family'] = df['SibSp'].map(lambda s : 1 if s > 1 else 0)
```

```
    status('family')
```

[85]:
```
process_family()
```

```
Processing family : ok
```

[86]:
```
df
```

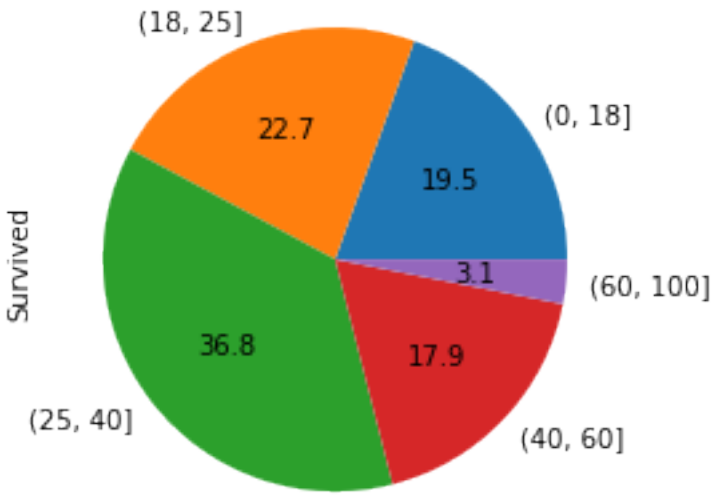
[86]:
```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           …       …     …
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                  Name     Sex   Age  SibSp  \
0                              Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                               Heikkinen, Miss. Laina  female  26.0      0
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                             Allen, Mr. William Henry    male  35.0      0
..                                                 …     …   …    …
886                             Montvila, Rev. Juozas    male  27.0      0
887                      Graham, Miss. Margaret Edith  female  19.0      0
888          Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                             Behr, Mr. Karl Howell    male  26.0      0
890                               Dooley, Mr. Patrick    male  32.0      0

     Parch            Ticket     Fare Cabin Embarked  FamilySize  family
0        0         A/5 21171   7.2500   NaN        S           2       0
1        0          PC 17599  71.2833   C85        C           2       0
2        0  STON/O2. 3101282   7.9250   NaN        S           1       0
3        0            113803  53.1000  C123        S           2       0
4        0            373450   8.0500   NaN        S           1       0
..     …               …      …   …      …         …     …
886      0            211536  13.0000   NaN        S           1       0
887      0            112053  30.0000   B42        S           1       0
888      2        W./C. 6607  23.4500   NaN        S           4       0
889      0            111369  30.0000  C148        C           1       0
890      0            370376   7.7500   NaN        Q           1       0
```

```
[891 rows x 14 columns]
```

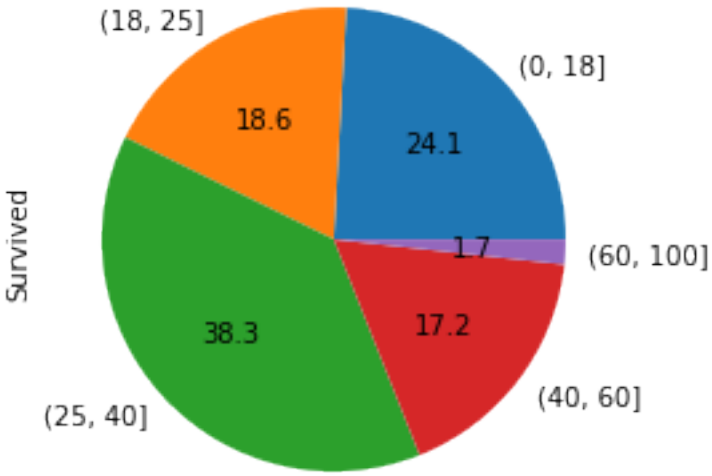
5 What are the age groups with more and less survivability?

Most survivability is between 25-40. With less was from 60-100

[87]:
```
group_by_age = pd.cut(df["Age"], [0,18,25,40,60,100])

age_grouping = df.groupby(group_by_age)['Survived'].count()

(age_grouping/age_grouping.sum()).plot(kind='pie',autopct='%.1f')

plt.show()
```



[88]:
```
group_by_age = pd.cut(df["Age"], [0,18,25,40,60,100])

age_grouping = df.groupby(group_by_age)['Survived'].sum()

(age_grouping/age_grouping.sum()).plot(kind='pie',autopct='%.1f')

plt.show()
```



[ ]:

5.1 da = df.drop('Ticket', 1)

da = da.drop('Cabin', 1)

da = da.dropna()

da.count()

5.2 How good is the following logistic regression model? Look at the R2 and p-values and conclude regarding both the quality of the model and the significance of each variable

The Port where they embarked and the parantel arch dont really affect the survivability. The R-squared is low thus the quality is only average is apparently low.

[89]:
```
import patsy

df_train=da.iloc[0:600,:]
df_test = da.iloc[0:112,:]
#formula = 'Survived ~ C(Pclass) + C(Sex) + Age + SibSp + C(Embarked) + Parch'
formula = 'Survived ~ C(Pclass) + C(Sex) + Age + SibSp'

y_train,x_train = patsy.dmatrices(formula,␣
 ↪data=df_train,return_type='dataframe')

y_test,x_test = patsy.dmatrices(formula, data=df_test,return_type='dataframe')
```

[90]:
```
import statsmodels.api as sm

model = sm.Logit(y_train,x_train)

res = model.fit()

res.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.451793
         Iterations 6
```

[90]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:               Survived   No. Observations:                  600
Model:                          Logit   Df Residuals:                      594
Method:                           MLE   Df Model:                            5
Date:                Thu, 21 Apr 2022   Pseudo R-squ.:                  0.3307
Time:                        17:36:49   Log-Likelihood:                -271.08
converged:                       True   LL-Null:                       -404.99
Covariance Type:            nonrobust   LLR p-value:                 8.172e-56
==============================================================================
==
                     coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
--
Intercept          4.1050      0.479      8.575      0.000       3.167
5.043
C(Pclass)[T.2]    -1.2971      0.306     -4.242      0.000      -1.896
-0.698
C(Pclass)[T.3]    -2.5739      0.305     -8.433      0.000      -3.172
-1.976
C(Sex)[T.male]    -2.5808      0.235    -10.996      0.000      -3.041
-2.121
Age               -0.0401      0.009     -4.549      0.000      -0.057
-0.023
SibSp             -0.3691      0.130     -2.840      0.005      -0.624
-0.114
==============================================================================
==
"""
```

6 A few examples of estimation versus real value... What is the estimation and what is the real value?

Left is real value, right is the estimation. First few and last few match but some miss. The model might be considered good because of accuracy

[91]:
```
model = sm.Logit(y_test,x_test)

res = model.fit()

y_pred = res.predict(x_test)

np.stack((y_test['Survived'],y_pred),1)
```

```
Optimization terminated successfully.
         Current function value: 0.399465
         Iterations 7
```

[91]:
```
array([[0.        , 0.04676248],
       [1.        , 0.79972164],
       [1.        , 0.68611342],
       [1.        , 0.816109  ],
       [0.        , 0.05421768],
       [0.        , 0.1313146 ],
       [0.        , 0.02826012],
       [1.        , 0.67847991],
       [1.        , 0.88836231],
       [1.        , 0.7197418 ],
       [1.        , 0.78479278],
       [0.        , 0.08860732],
       [0.        , 0.02625069],
       [0.        , 0.7693435 ],
       [1.        , 0.77623162],
       [0.        , 0.01550102],
       [0.        , 0.49808463],
       [0.        , 0.20166803],
       [1.        , 0.2073974 ],
       [1.        , 0.76303502],
       [1.        , 0.27413194],
       [0.        , 0.39535298],
       [1.        , 0.43679584],
       [0.        , 0.07602777],
       [0.        , 0.19839508],
       [0.        , 0.07816027],
       [0.        , 0.16975713],
       [0.        , 0.11101855],
       [0.        , 0.08580428],
       [0.        , 0.45923094],
```

```
[1.        , 0.64359851],
[0.        , 0.4195529 ],
[0.        , 0.8342853 ],
[1.        , 0.92139846],
[1.        , 0.73663003],
[0.        , 0.61067529],
[0.        , 0.01303098],
[0.        , 0.08580428],
[1.        , 0.73050282],
[1.        , 0.82431716],
[0.        , 0.09306561],
[1.        , 0.91991531],
[0.        , 0.06722626],
[1.        , 0.91614369],
[0.        , 0.00617062],
[0.        , 0.08308183],
[0.        , 0.1010122 ],
[0.        , 0.02638888],
[1.        , 0.89655001],
[0.        , 0.09149276],
[1.        , 0.20498826],
[0.        , 0.0225494 ],
[0.        , 0.21921405],
[0.        , 0.12633294],
[0.        , 0.29258298],
[0.        , 0.04086929],
[1.        , 0.05989739],
[0.        , 0.07537997],
[1.        , 0.45696739],
[1.        , 0.65501239],
[0.        , 0.08308183],
[1.        , 0.06613048],
[0.        , 0.27413194],
[1.        , 0.92969819],
[1.        , 0.21327845],
[0.        , 0.05713617],
[1.        , 0.66499186],
[0.        , 0.07787147],
[0.        , 0.06613048],
[0.        , 0.08860732],
[0.        , 0.09785887],
[0.        , 0.04086929],
[0.        , 0.02402836],
[0.        , 0.0766992 ],
[1.        , 0.3105227 ],
[1.        , 0.87904141],
[0.        , 0.12408786],
```

```
[0.        , 0.67074984],
[0.        , 0.32579981],
[0.        , 0.05794483],
[0.        , 0.01541894],
[0.        , 0.06833886],
[1.        , 0.72273987],
[0.        , 0.04904845],
[0.        , 0.16207488],
[0.        , 0.63954942],
[0.        , 0.08308183],
[0.        , 0.59380529],
[0.        , 0.75006496],
[0.        , 0.08580428],
[0.        , 0.01615582],
[0.        , 0.14452629],
[0.        , 0.30303368],
[0.        , 0.30424628],
[0.        , 0.10812351],
[0.        , 0.1299443 ],
[1.        , 0.88454662],
[0.        , 0.1313146 ],
[1.        , 0.06521549],
[1.        , 0.07787147],
[0.        , 0.03874756],
[0.        , 0.05794483],
[0.        , 0.08860732],
[0.        , 0.36097755],
[1.        , 0.82431716],
[0.        , 0.2643    ],
[0.        , 0.27822068],
[1.        , 0.93506673],
[0.        , 0.12962291],
[0.        , 0.10066171],
[0.        , 0.30303368],
[1.        , 0.71562791]])
```

7 What does precision and recall for each case (survived/dd not survive) tell us?

73 of the test cases of people that died where classified by the model as having died, only 2 of the 75 dead were classified as not having died 16 of the 37 that died were wrongly classified as not having survived, and 21 properly classified. The model seems to hav ean easier time classifing the not survives than the survives

No precision e recall o valor que nao esta muto bom e o 57%, Dos que sobreviveram apenas 57% foram bem classificados. Precision = True positive/(True positive + False Positive) Recall = True positive/(True positive + False Negative) The ones that survived

[92]:
```
y_pred_flag=y_pred>0.7

print(pd.crosstab(y_train.Survived,y_pred_flag))

import sklearn.metrics as smet

print( smet.classification_report(y_test,y_pred_flag)  )
```

```
col_0     False  True
Survived
0.0          72     3
1.0          16    21
              precision    recall  f1-score   support

         0.0       0.82      0.96      0.88        75
         1.0       0.88      0.57      0.69        37

    accuracy                           0.83       112
   macro avg       0.85      0.76      0.79       112
weighted avg       0.84      0.83      0.82       112
```

8 Can you do better than this by choosing carefully the variables to use? What would you try? Try it once....

No Real way of improving by changing variables, the data dictate the correlations, whichever they are. Removing the High-P-Value varabiles does very little seens as they already had no impact.

9 THE END