Sinking of Titanic

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

Do a complete analysis on what sorts of people were likely to survive.

In [1]:
```
import pandas as pd
import numpy as np

# import plotting libraries
import matplotlib
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
%matplotlib inline

import seaborn as sns
sns.set(style="white", color_codes=True)
sns.set(font_scale=1.5)

# import libraries for model validation
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

# import libraries for metrics and reporting
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn import metrics
from sklearn.metrics import classification_report
```

In [3]:
```
cd E:\DATA SCIENCE -SIMPLILEARN\MACHINE LEARNING\Day3
```

```
E:\DATA SCIENCE -SIMPLILEARN\MACHINE LEARNING\Day3
```

In [4]:
```
df_train = pd.read_csv("train.csv")
df_train.shape
```

Out[4]:
```
(891, 12)
```

In [5]:
```
# see distinct values in the Sex column
df_train.Survived.value_counts()
```

Out[5]:
```
0    549
1    342
Name: Survived, dtype: int64
```

In [6]:
```
# see distinct values in the Sex column
df_train.Sex.value_counts()
```

Out[6]:
```
male      577
female    314
Name: Sex, dtype: int64
```

In [7]:
```
# see distinct values in the Embarked column
df_train.Embarked.value_counts()
```

Out[7]:
```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

In [8]:
```
# Checking for missing values
# It's easy to check for missing values by calling the isnull() method, and
# the sum() method off of that, to return a tally of all the True values that are returned
# by the isnull() method.

df_train.isnull().sum()
```

Out[8]:
```
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 [9]:
```
df_train.shape
```

Out[9]:
```
(891, 12)
```

In [9]:
```
# there are only 891 rows in the titanic data frame. Cabin is almost all missing values,
# so we can drop that variable completely,

# but what about age?
# Age seems like a relevant predictor for survival right? We'd want to keep the variables,
# but it has 177 missing values.

# Need to find a way to approximate for those missing values!
```

In [10]:
```
# drop all the variables that aren't relevant for predicting survival.
# We should at least keep the following:

# Survived - This variable is obviously relevant.
# Pclass   - Does a passenger's class on the boat affect their survivability?
# Sex      - Could a passenger's gender impact their survival rate?
# Age      - Does a person's age impact their survival rate?
# SibSp    - Does the number of relatives on the boat (that are siblings or a spouse)
#            affect a person survivability? Probability
# Parch    - Does the number of relatives on the boat (that are children or parents)
#            affect a person survivability? Probability
# Fare     - Does the fare a person paid effect his survivability? Maybe - let's keep it.
# Embarked - Does a person's point of embarkation matter?
#            It depends on how the boat was filled... Let's keep it.
```

In [11]:
```
# What about a person's name, ticket number, and passenger ID number?
# They're irrelavant for predicting survivability.
# And as you recall, the cabin variable is almost all missing values,
# so we can just drop all of these.
```

In [10]:
```
df_train = df_train.drop(['PassengerId','Name','Ticket','Cabin'], axis=1)
df_train.head()
```

Out[10]:

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

In [11]:
```
# now we need to deal with the missing values in the age variable.

# Speaking roughly, we could say that the younger a passenger is, the more likely it is for them
# to be in 3rd class. The older a passenger is, the more likely it is for them to be in 1st class. So there is a loose relationship between these variables. So, let's write a function that approximates a passengers age, based on their class. From the box plot, it looks like the average age of 1st class passengers is about 37, 2nd class passengers is 29, and 3rd class pasengers is 24.

# find each null value in the Age variable and for each null, checks the value of the Pclass
# and assigns an age value according to the average age of passengers in that class.

def age_approx(cols):
    Age    = cols[0]
    Pclass = cols[1]

    if pd.isnull(Age):
        if Pclass == 1:
            return 37
        elif Pclass == 2:
            return 29
        else:
            return 24
    else:
        return Age
```

In [14]:
```
# avg age per class
# df_train[df_train['Pclass']== 1]['Age'].mean()
# df_train[df_train['Pclass']== 2]['Age'].mean()
# df_train[df_train['Pclass']== 3]['Age'].mean()
```

In [12]:
```
df_train.groupby(['Pclass']).mean()
```

Out[12]:

Pclass	Survived	Age	SibSp	Parch	Fare
1	0.629630	38.233441	0.416667	0.356481	84.154687
2	0.472826	29.877630	0.402174	0.380435	20.662183
3	0.242363	25.140620	0.615071	0.393075	13.675550

In [13]:
```
df_train['Age'] = df_train[['Age', 'Pclass']].apply(age_approx, axis=1)
```

In [14]:
```
# check for null again
df_train.isnull().sum()
```

Out[14]:
```
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    2
dtype: int64
```

In [18]:
```
# There are 2 null values in the embarked variable. We can drop those 2 records without
# loosing too much important information from our dataset, so we will do that.
```

In [15]:
```
df_train.dropna(inplace=True)
df_train.isnull().sum()
```

Out[15]:
```
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    0
dtype: int64
```

In [20]:
```
# Converting categorical variables to a dummy indicators **********
```

In [16]:
```
# object signifies they are of categorical/string type data
df_train.dtypes
```

Out[16]:
```
Survived      int64
Pclass        int64
Sex          object
Age         float64
SibSp         int64
Parch         int64
Fare        float64
Embarked     object
dtype: object
```

In [17]:
```
#pd.get_dummies(df_train['Sex'], drop_first=True)
df_train_dummied = pd.get_dummies(df_train, columns=["Sex"])
```

In [18]:
```
df_train_dummied = pd.get_dummies(df_train_dummied, columns=["Embarked"])
```

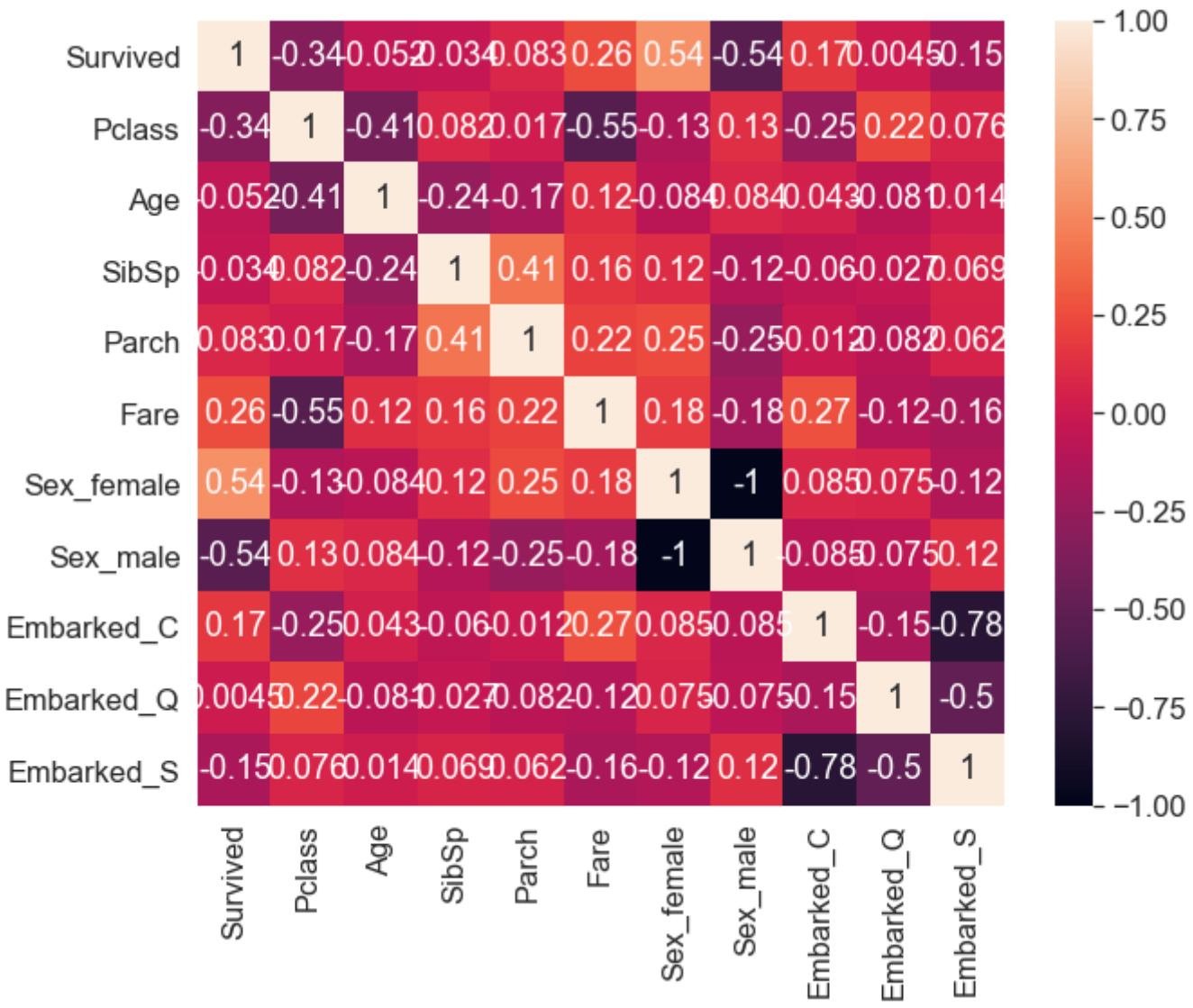
In [19]:
```
df_train_dummied.head()
```

Out[19]:

	Survived	Pclass	Age	SibSp	Parch	Fare	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	0	3	22.0	1	0	7.2500	0	1	0	0	1
1	1	1	38.0	1	0	71.2833	1	0	1	0	0
2	1	3	26.0	0	0	7.9250	1	0	0	0	1
3	1	1	35.0	1	0	53.1000	1	0	0	0	1
4	0	3	35.0	0	0	8.0500	0	1	0	0	1

In [23]:
```
# Checking for independence between features
plt.figure(figsize=(10,8))
sns.heatmap(df_train_dummied.corr(),annot=True)
```

Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa17eb70b88>
```



In [24]:
```
used_features =[
    "Pclass",
    "Age",
    "SibSp",
    "Parch",
    "Sex_female",
    "Sex_male",
    "Embarked_C",
    "Embarked_Q",
    "Embarked_S"
]

X = df_train_dummied[used_features].values
y = df_train_dummied['Survived']
```

In [25]:
```
# Split dataset in training and test datasets
# X_train, X_test = train_test_split(df_train, test_size=0.5, random_state=int(time.time()))
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=11)
```

In [26]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(622, 9)
(267, 9)
(622,)
(267,)
```

In [27]:
```
# Instantiate the classifier
LogReg = LogisticRegression()
```

In [28]:
```
# Train classifier
LogReg.fit(X_train, y_train)
```

```
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs failed to
converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[28]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [29]:
```
y_pred = LogReg.predict(X_test)
```

In [30]:
```
metrics.confusion_matrix(y_test, y_pred)
```

Out[30]:
```
array([[143,  18],
       [ 23,  83]], dtype=int64)
```

In [31]:
```
metrics.accuracy_score(y_test, y_pred)
```

Out[31]:
```
0.846441947565543
```

In [32]:
```
len(X_test)
```

Out[32]:
```
267
```

In [33]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.86      0.89      0.87       161
           1       0.82      0.78      0.80       106

    accuracy                           0.85       267
   macro avg       0.84      0.84      0.84       267
weighted avg       0.85      0.85      0.85       267
```

In [34]:
```
LogReg.coef_
```

Out[34]:
```
array([[-1.18878588, -0.03958562, -0.26115394, -0.14558233,  1.14457962,
        -1.22104432,  0.19607131,  0.05331804, -0.32585404]])
```

In [35]:
```
LogReg.intercept_
```

Out[35]:
```
array([4.03382543])
```

In [36]:
```
df_train_dummied[used_features].columns
```

Out[36]:
```
Index(['Pclass', 'Age', 'SibSp', 'Parch', 'Sex_female', 'Sex_male',
       'Embarked_C', 'Embarked_Q', 'Embarked_S'],
      dtype='object')
```

In [37]:
```
LogReg.predict_proba(X_test)
```

Out[37]:
```
array([[0.36557929, 0.63442071],
       [0.9347247 , 0.0652753 ],
       [0.54153578, 0.45846422],
       [0.48205791, 0.51794209],
       [0.84286381, 0.15713619],
       [0.85419049, 0.14580951],
       [0.566731  , 0.433269  ],
       [0.13991442, 0.86008558],
       [0.06537171, 0.93462829],
       [0.83893341, 0.16106659],
       [0.5615648 , 0.4384352 ],
       [0.86194716, 0.13805284],
       [0.83893341, 0.16106659],
       [0.15727175, 0.84272825],
       [0.22893503, 0.77106497],
       [0.8513304 , 0.1486696 ],
       [0.87118391, 0.12881609],
       [0.56934731, 0.43065269],
       [0.07194108, 0.92805892],
       [0.91262936, 0.08737064],
       [0.40251806, 0.59748194],
       [0.62863498, 0.37136502],
       [0.87110099, 0.12889901],
       [0.80665017, 0.19334983],
       [0.88385828, 0.11614172],
       [0.89729795, 0.10270205],
       [0.88385828, 0.11614172],
       [0.46253085, 0.53746915],
       [0.69021273, 0.30978727],
       [0.93468424, 0.06531576],
       [0.07871988, 0.92128012],
       [0.72811204, 0.27188796],
       [0.90093178, 0.09906822],
       [0.58181705, 0.41818295],
       [0.88385828, 0.11614172],
       [0.94381063, 0.05618937],
       [0.36039454, 0.63960546],
       [0.92048697, 0.07951303],
       [0.89915113, 0.10084887],
       [0.91262936, 0.08737064],
       [0.06783234, 0.93216766],
       [0.96816951, 0.03183049],
       [0.86194716, 0.13805284],
       [0.29795939, 0.70204061],
       [0.54450062, 0.45549938],
       [0.28726248, 0.71273752],
       [0.32842648, 0.67157352],
       [0.76979002, 0.23020998],
       [0.48875846, 0.51124154],
       [0.1069954 , 0.8930046 ],
       [0.81723266, 0.18276734],
       [0.71430925, 0.28569075],
       [0.25171576, 0.74828424],
       [0.68142131, 0.31857869],
       [0.4321922 , 0.5678078 ],
       [0.82538133, 0.17461867],
       [0.125874  , 0.874126  ],
       [0.83893341, 0.16106659],
       [0.86194716, 0.13805284],
       [0.89915113, 0.10084887],
       [0.20317267, 0.79682733],
       [0.20317267, 0.79682733],
       [0.11296294, 0.88703706],
       [0.88536611, 0.11463389],
       [0.59790137, 0.40209863],
       [0.92164623, 0.07835377],
       [0.41207393, 0.58792607],
       [0.15134772, 0.84865228],
       [0.05290945, 0.94709055],
       [0.81869935, 0.18130065],
       [0.88221139, 0.11778861],
       [0.86194716, 0.13805284],
       [0.07754384, 0.92245616],
       [0.26133976, 0.73866024],
       [0.17873754, 0.82126246],
       [0.92980412, 0.07019588],
       [0.30223316, 0.69776684],
       [0.39458734, 0.60541266],
       [0.16196479, 0.83803521],
       [0.42542783, 0.57457217],
       [0.83893341, 0.16106659],
       [0.06174421, 0.93825579],
       [0.03727012, 0.96272988],
       [0.05122468, 0.94877532],
       [0.05559473, 0.94440527],
       [0.88385828, 0.11614172],
       [0.84365724, 0.15634276],
       [0.88385828, 0.11614172],
```

```
[0.87110099, 0.12889901],
[0.84625018, 0.15374982],
[0.72918919, 0.27081081],
[0.73858515, 0.26141485],
[0.75852769, 0.24147231],
[0.88385828, 0.11614172],
[0.32842648, 0.67157352],
[0.87973255, 0.12026745],
[0.05771007, 0.94228993],
[0.91573467, 0.08426533],
[0.90942093, 0.09057907],
[0.36140482, 0.63859518],
[0.36219048, 0.63780952],
[0.84901109, 0.15098891],
[0.07475918, 0.92524082],
[0.91361941, 0.08638059],
[0.90013343, 0.09986657],
[0.33039782, 0.66960218],
[0.48524453, 0.51475547],
[0.81869935, 0.18130065],
[0.61893467, 0.38106533],
[0.87150106, 0.12849894],
[0.10475753, 0.89524247],
[0.89174173, 0.10825827],
[0.16258978, 0.83741022],
[0.31198827, 0.68801173],
[0.4753638 , 0.5246362 ],
[0.0467944 , 0.9532056 ],
[0.87534816, 0.12465184],
[0.94799447, 0.05200553],
[0.81869935, 0.18130065],
[0.86574284, 0.13425716],
[0.83893341, 0.16106659],
[0.91262936, 0.08737064],
[0.9244576 , 0.0755424 ],
[0.85716885, 0.14283115],
[0.91873944, 0.08126056],
[0.6828855 , 0.3171145 ],
[0.88385828, 0.11614172],
[0.74068037, 0.25931963],
[0.87728835, 0.12271165],
[0.31777523, 0.68222477],
[0.0378465 , 0.9621535 ],
[0.52182504, 0.47817496],
[0.27036608, 0.72963392],
[0.64711585, 0.35288415],
[0.5010467 , 0.4989533 ],
[0.32842648, 0.67157352],
[0.81723266, 0.18276734],
[0.81290599, 0.18709401],
[0.13484627, 0.86515373],
[0.18608146, 0.81391854],
[0.74615574, 0.25384426],
[0.81869935, 0.18130065],
[0.42926259, 0.57073741],
[0.41207393, 0.58792607],
[0.9244576 , 0.0755424 ],
[0.88882769, 0.11117231],
[0.93946147, 0.06053853],
[0.83893341, 0.16106659],
[0.125874  , 0.874126  ],
[0.95682246, 0.04317754],
[0.38111507, 0.61888493],
[0.87548091, 0.12451909],
[0.42250507, 0.57749493],
[0.0775967 , 0.9224033 ],
[0.88373328, 0.11626672],
[0.73087021, 0.26912979],
[0.07157924, 0.92842076],
[0.91262936, 0.08737064],
[0.17168924, 0.82831076],
[0.06694836, 0.93305164],
[0.86659048, 0.13340952],
[0.63951579, 0.36048421],
[0.88536611, 0.11463389],
[0.38696447, 0.61303553],
[0.1174177 , 0.8825823 ],
[0.88385828, 0.11614172],
[0.84719918, 0.15280082],
[0.93645896, 0.06354104],
[0.23001274, 0.76998726],
[0.80759902, 0.19240098],
[0.92980412, 0.07019588],
[0.16127843, 0.83872157],
[0.52966928, 0.47033072],
[0.87118391, 0.12881609],
[0.88385828, 0.11614172],
[0.91573467, 0.08426533],
```

```
[0.81869935, 0.18130065],
[0.54153578, 0.45846422],
[0.64403158, 0.35596842],
[0.125874  , 0.874126  ],
[0.81869935, 0.18130065],
[0.1174177 , 0.8825823 ],
[0.58702852, 0.41297148],
[0.87096426, 0.12903574],
[0.89400287, 0.10599713],
[0.88385828, 0.11614172],
[0.24140107, 0.75859893],
[0.41674781, 0.58325219],
[0.44556692, 0.55443308],
[0.9026844 , 0.0973156 ],
[0.06615568, 0.93384432],
[0.82120536, 0.17879464],
[0.83893341, 0.16106659],
[0.2029757 , 0.7970243 ],
[0.88385828, 0.11614172],
[0.64638486, 0.35361514],
[0.72348348, 0.27651652],
[0.27904919, 0.72095081],
[0.0378465 , 0.9621535 ],
[0.88385828, 0.11614172],
[0.8691616 , 0.1308384 ],
[0.97052096, 0.02947904],
[0.83893341, 0.16106659],
[0.355051  , 0.644949  ],
[0.88385828, 0.11614172],
[0.25510252, 0.74489748],
[0.94839974, 0.05160026],
[0.94978658, 0.05021342],
[0.55134648, 0.44865352],
[0.05949034, 0.94050966],
[0.92980412, 0.07019588],
[0.88385828, 0.11614172],
[0.48126266, 0.51873734],
[0.6025587 , 0.3974413 ],
[0.41335436, 0.58664564],
[0.69835622, 0.30164378],
[0.89174173, 0.10825827],
[0.76989191, 0.23010809],
[0.98798939, 0.01201061],
[0.69021273, 0.30978727],
[0.81869935, 0.18130065],
[0.20965667, 0.79034333],
[0.94438485, 0.05561515],
[0.72966963, 0.27033037],
[0.92756769, 0.07243231],
[0.44773756, 0.55226244],
[0.82223395, 0.17776605],
[0.063128  , 0.936872  ],
[0.36039454, 0.63960546],
[0.76085717, 0.23914283],
[0.69369308, 0.30630692],
[0.4361094 , 0.5638906 ],
[0.88385828, 0.11614172],
[0.88385828, 0.11614172],
[0.30738292, 0.69261708],
[0.59582408, 0.40417592],
[0.3962017 , 0.6037983 ],
[0.94587343, 0.05412657],
[0.16419327, 0.83580673],
[0.89174173, 0.10825827],
[0.98216798, 0.01783202],
[0.10945837, 0.89054163],
[0.94394991, 0.05605009],
[0.36195265, 0.63804735],
[0.62387764, 0.37612236],
[0.30156185, 0.69843815],
[0.92437255, 0.07562745],
[0.81869935, 0.18130065],
[0.95682246, 0.04317754],
[0.54153578, 0.45846422],
[0.07871988, 0.92128012],
[0.45270588, 0.54729412],
[0.75335414, 0.24664586],
[0.89174173, 0.10825827],
[0.24777036, 0.75222964],
[0.88385828, 0.11614172],
[0.20755013, 0.79244987],
[0.2804359 , 0.7195641 ],
[0.94167407, 0.05832593],
[0.88385828, 0.11614172],
[0.87548091, 0.12451909],
[0.7199293 , 0.2800707 ],
[0.88385828, 0.11614172],
[0.64886774, 0.35113226],
```

```
       [0.29971948, 0.70028052],
       [0.38837297, 0.61162703],
       [0.37855218, 0.62144782]])
```

In [ ]: