Machine Learning & The Titanic Data

By Souky, Jin, Julie, Cesur, and Emma
April 18, 2020

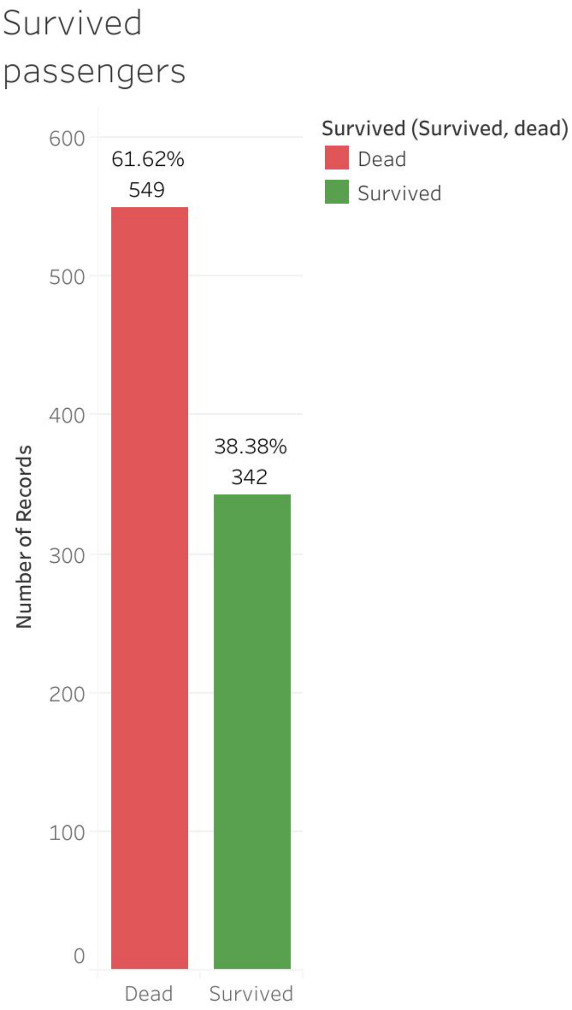
Agenda:

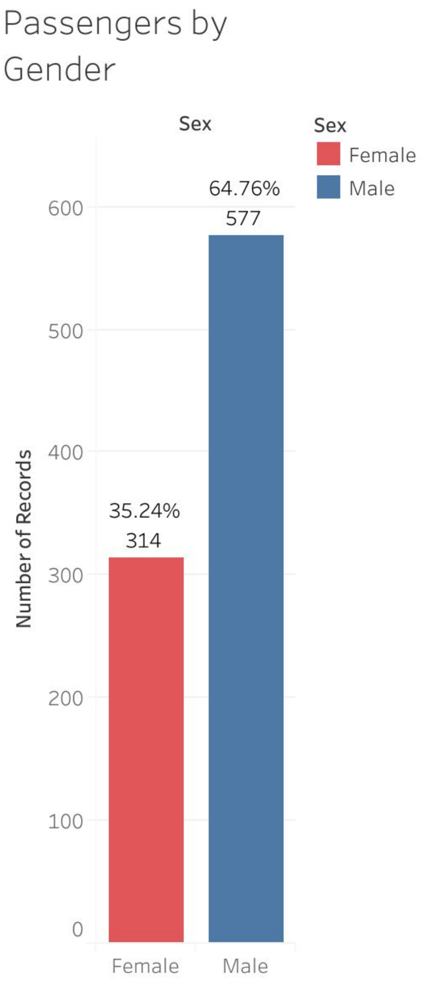
1. Tableau Graphics
2. Pandas Charts
3. Machine Learning Overview and Findings
4. Project Conclusions

Tableau & Pandas Storytelling

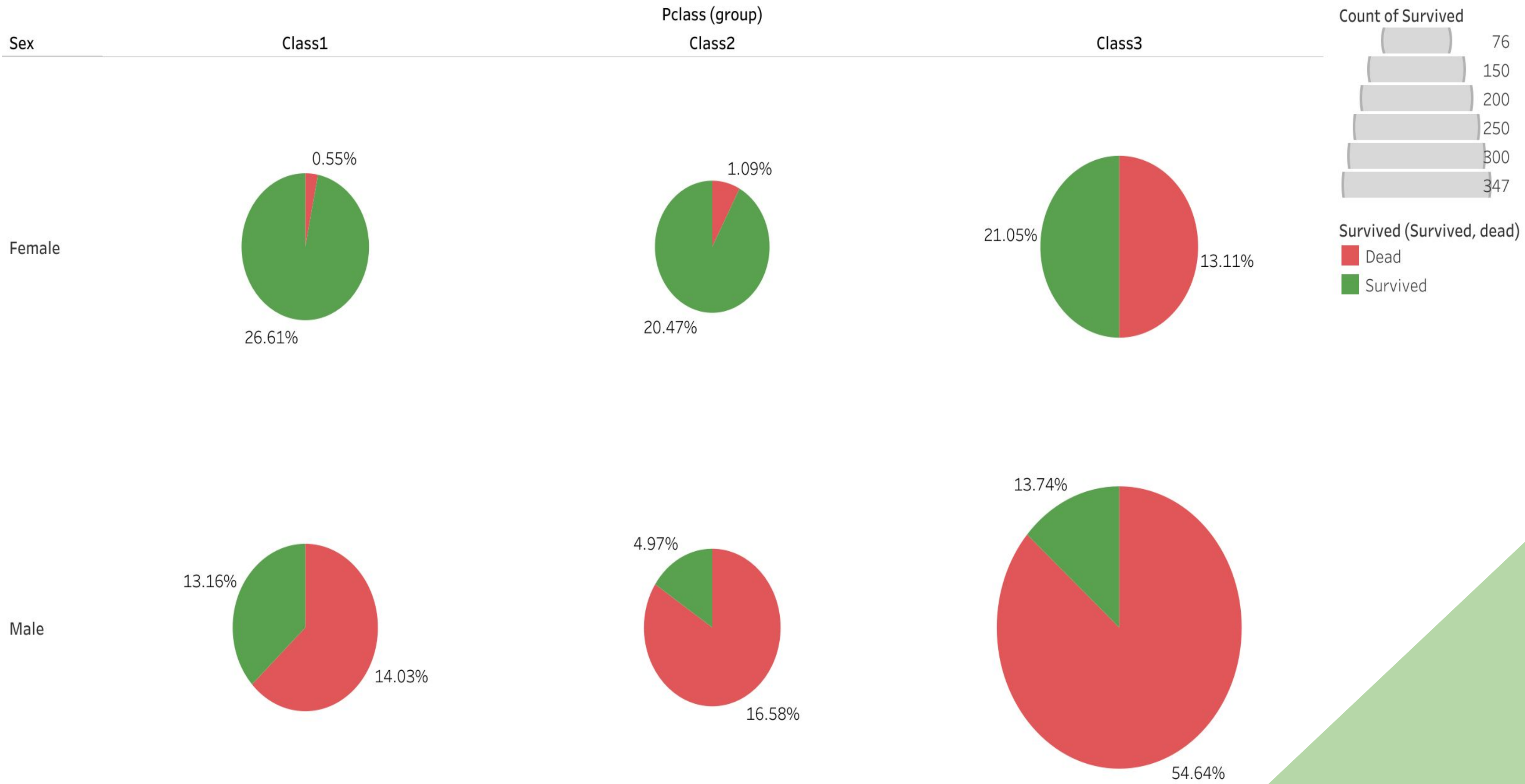
- How many people Survived and how many people died ?
- Survival Analysis by Gender .
- Survival Analysis by Class .
- Survival Analysis by Port Embarkation.
- Survival rate per each factors
- Passenger with which traits is likely to

Survived: Analysis by Gender

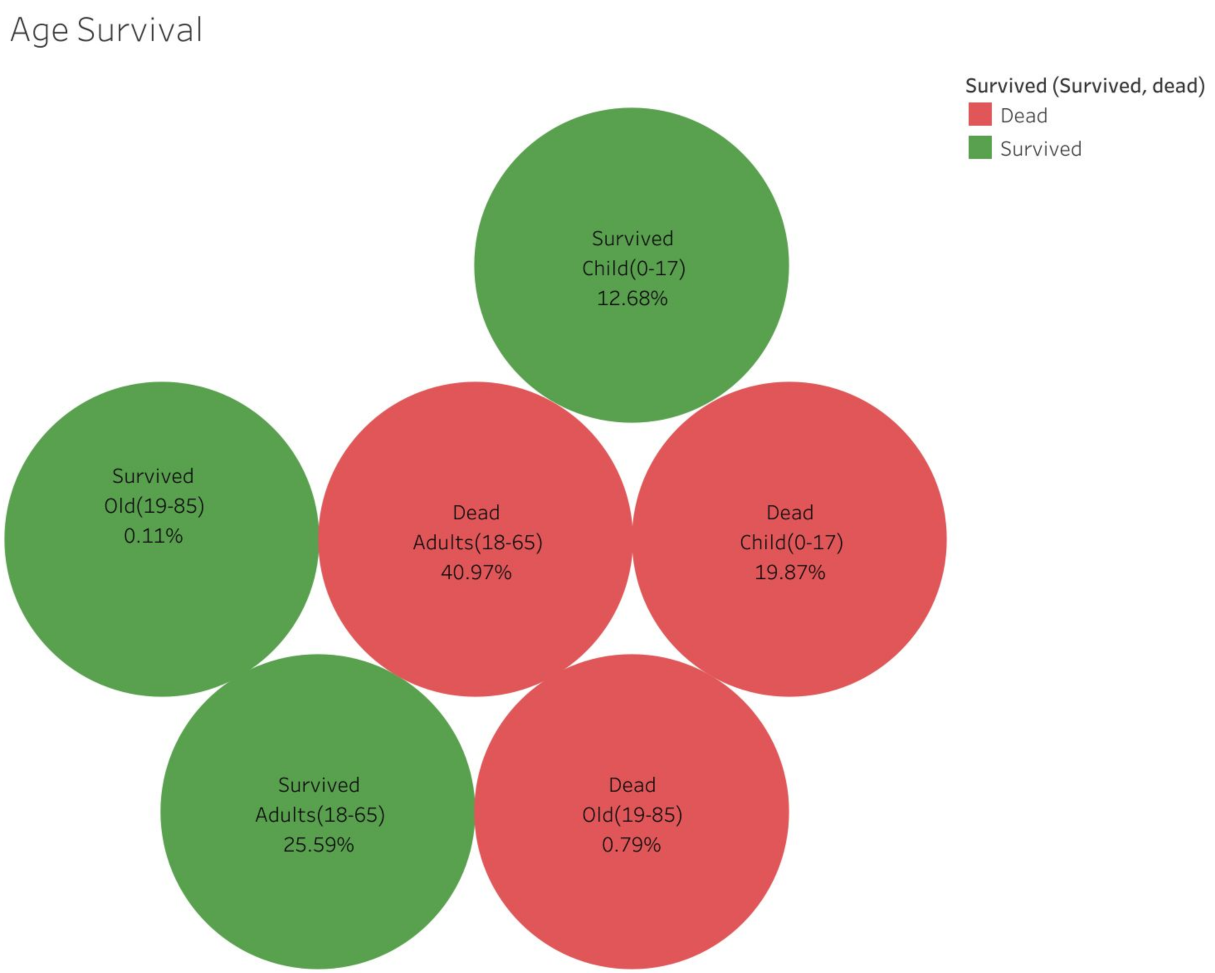


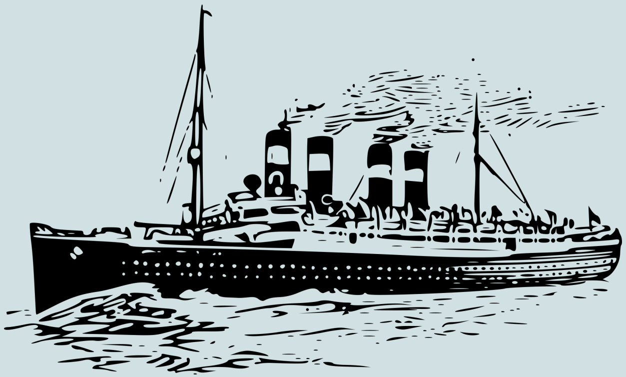


Survival Rate Passenger by Gender & Class



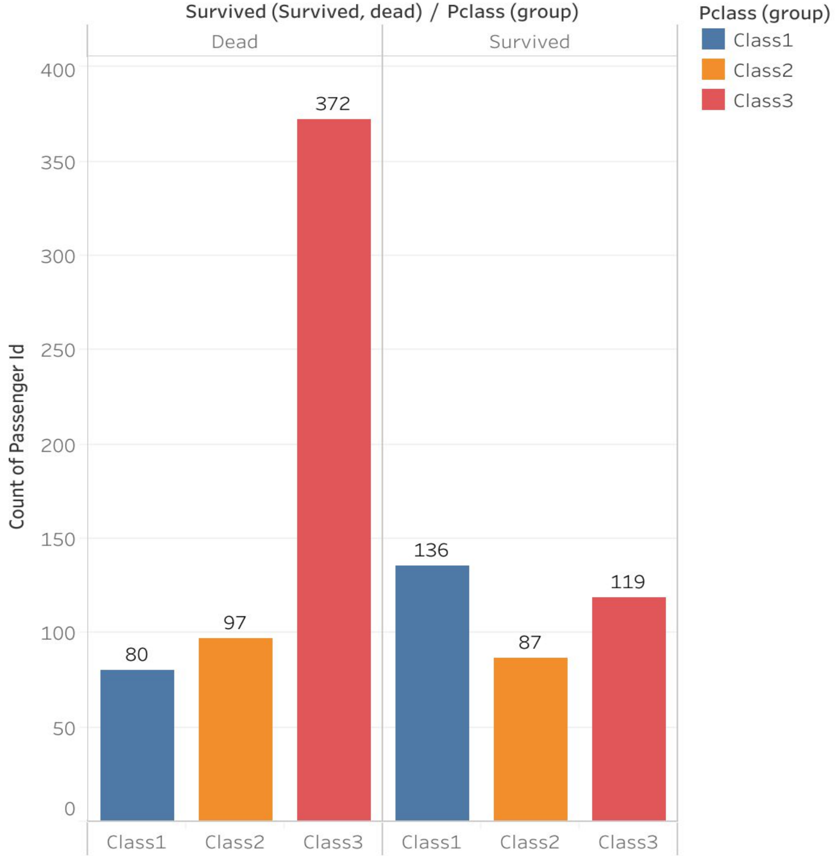
Survived: Age



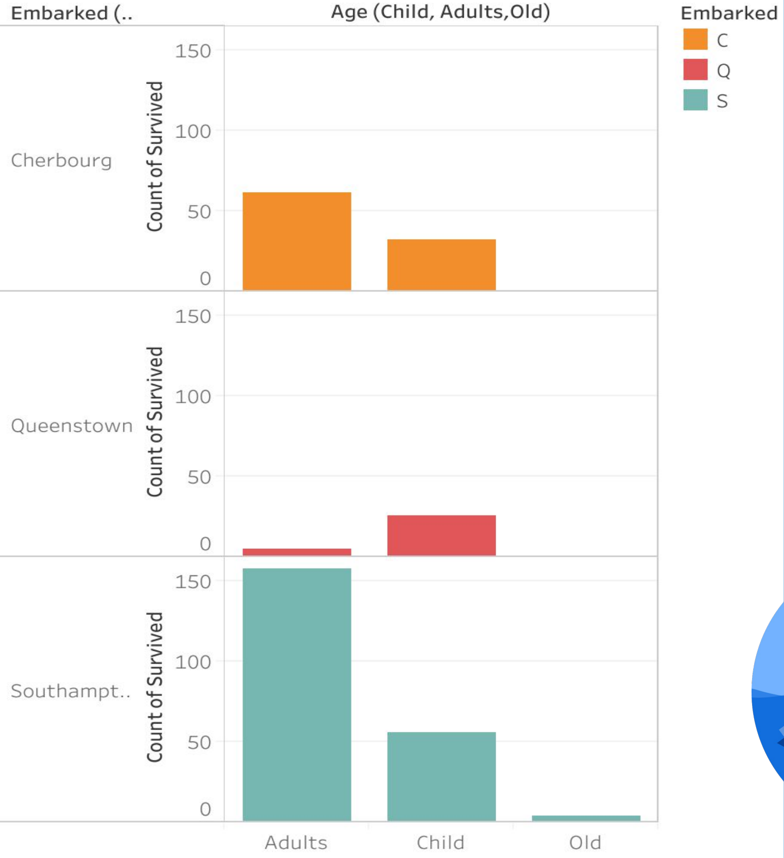


Survived by Class and Embarkation

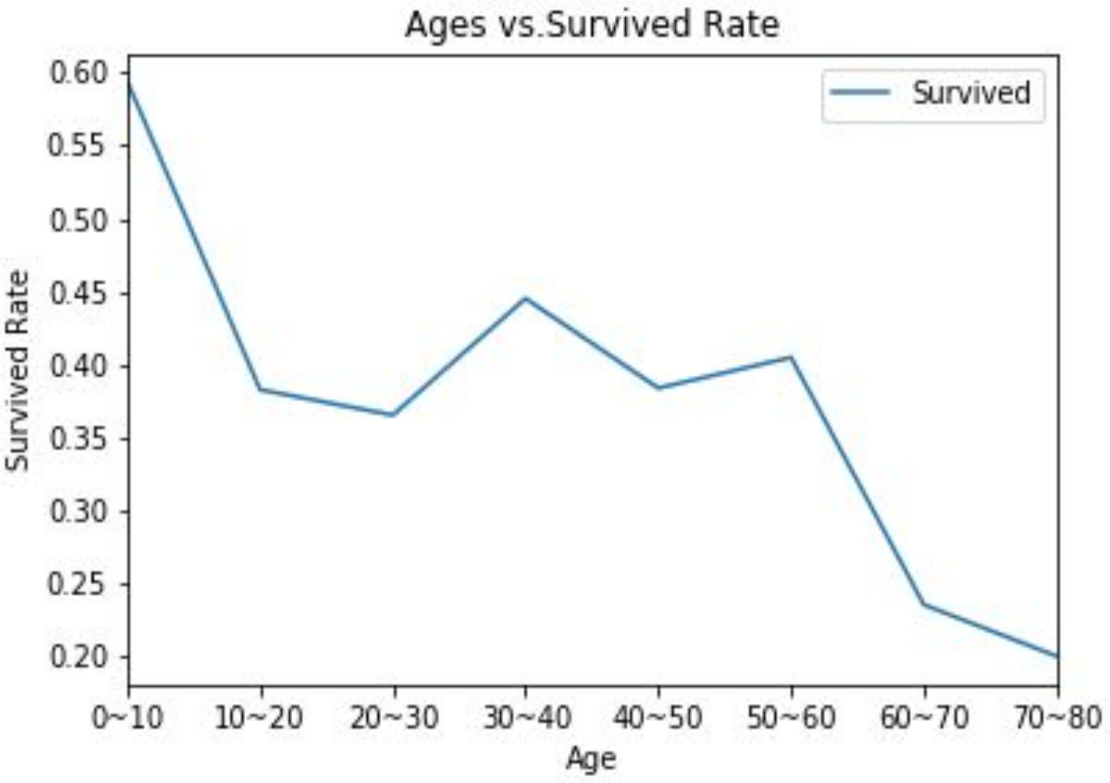
Survived by class

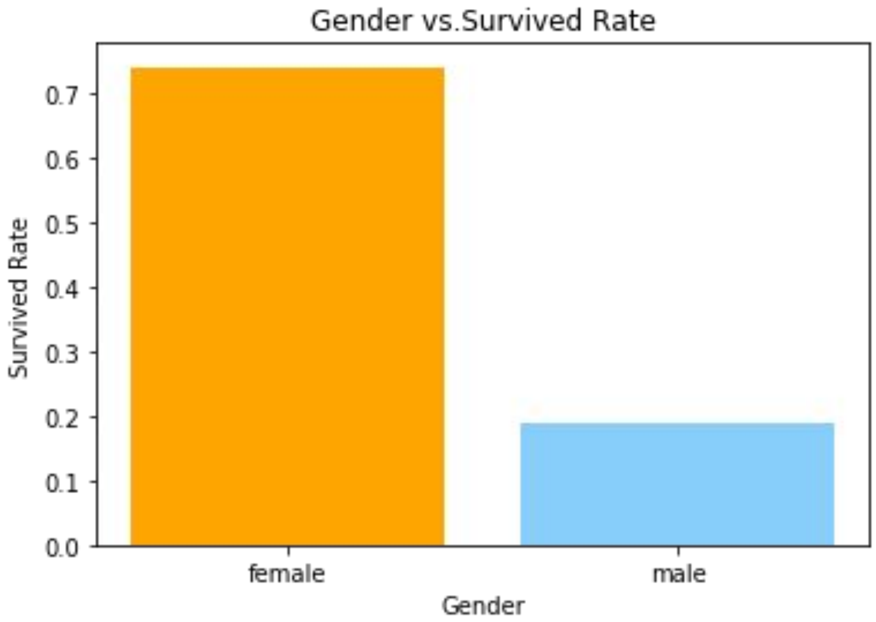


Survived by port of Embarkation

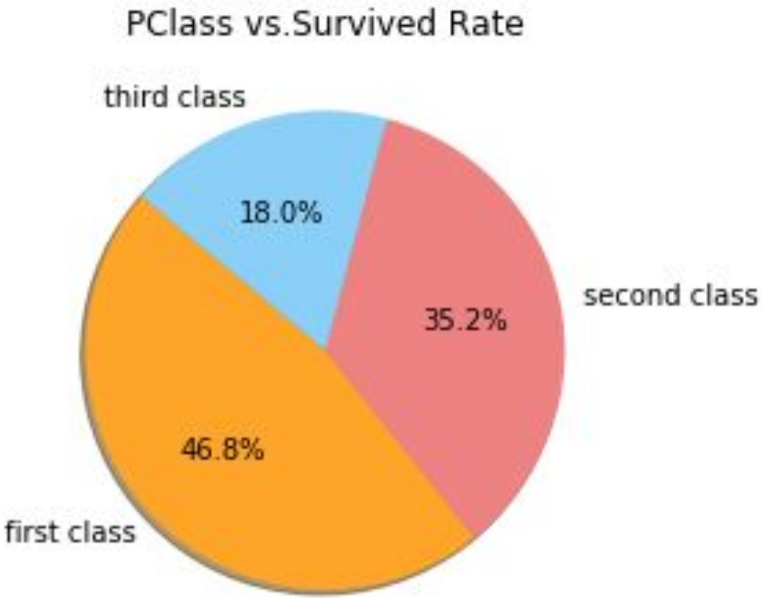


Pandas: Survived rate per factors





Gender vs.Survived Rate
Survived Rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
female
male
Gender



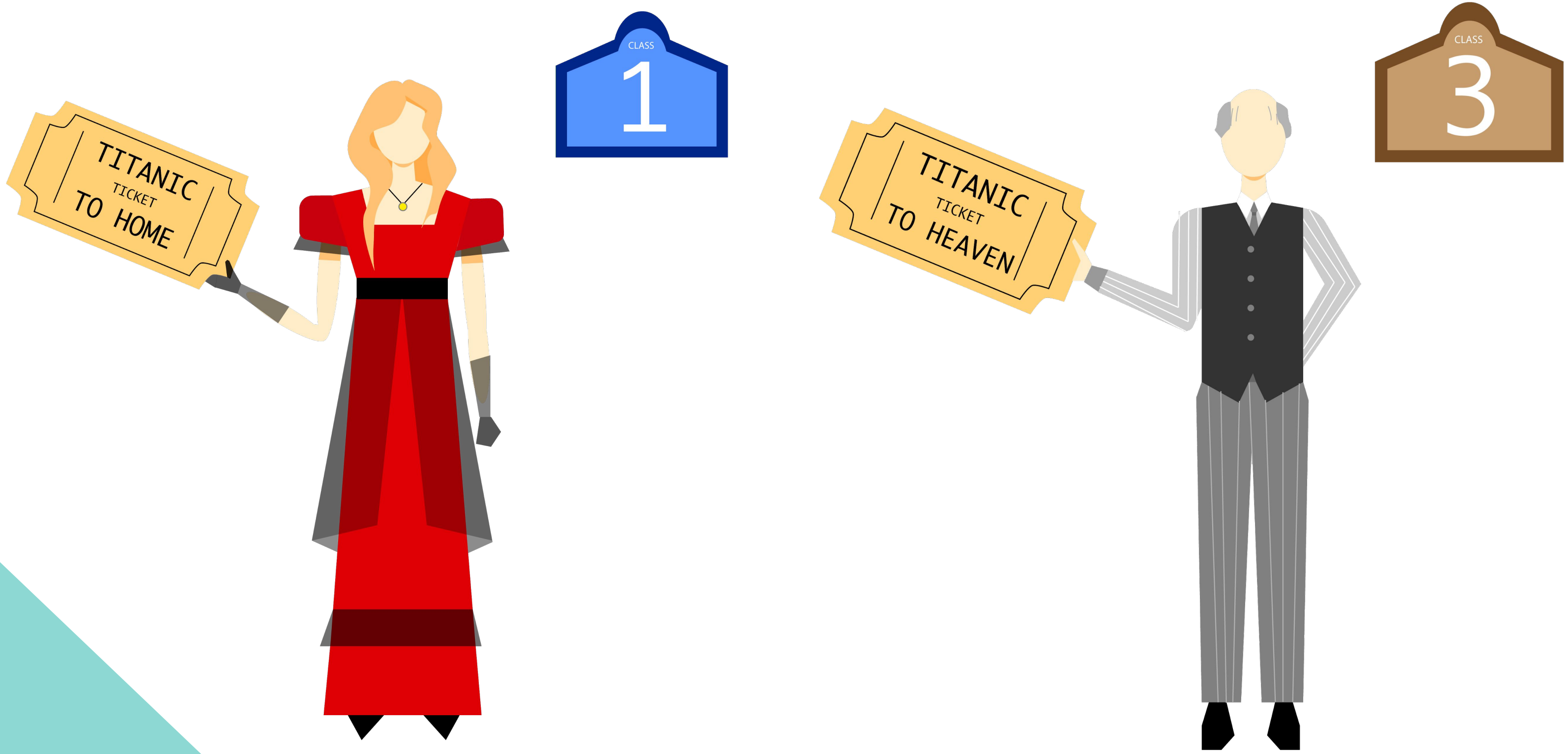
PClass vs.Survived Rate
third class
18.0%
second class
35.2%
first class
46.8%

Who is more likely to survive?

Survive%	Pclass	Sex	Age
100.0	1	female	10~20
100.0	1	female	30~40
100.0	1	female	50~60
100.0	1	female	60~70
100.0	1	male	0~10
100.0	2	female	0~10
100.0	2	female	10~20
100.0	2	male	0~10

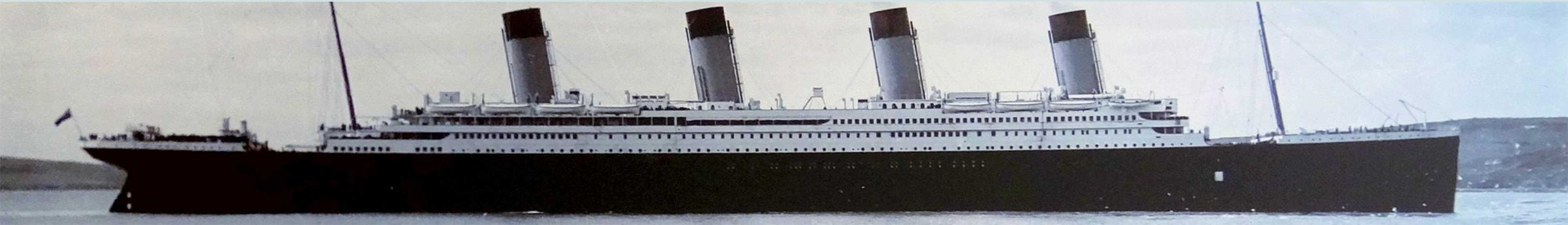
Survive%	Pclass	Sex	Age
0.0	1	male	60~70
0.0	2	male	20~30
0.0	2	male	50~60
0.0	3	female	40~50
0.0	3	male	50~60
0.0	3	male	60~70
0.0	3	male	70~80

The Conclusion of our Story



Our ML Objective

1. Determine the best machine learning algorithm for our Titanic dataset by utilizing different feature engineering and data cleaning methods
2. Have fun with machine learning
   a. undergo multiple trial and error tests to experiment with best case scenarios (playing with multiple algorithms and how the code affects accuracy)



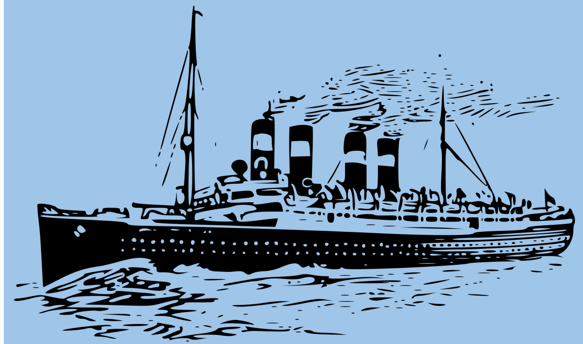
ML: Random Forest

`x_train : input_variables_values_training_datasets`

`y_train : target_variables_values_training_datasets`

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

	importance
feature	
Fare	0.297
Age	0.269
Sex	0.268
Pclass	0.084
SibSp	0.045



ML: Confusion Matrix and Comparing Algorithms (Logistic Regression, Decision Tree, and Random Forest)

Estimate Accuracy Scores

Random Forest: 98.2 %

Decision Tree: 98.2 %

Logistic Regression: 79.8 %

CORRECT	INCORRECT
Not Survived,	Not Survived
INCORRECT	**CORRECT**
Survived,	Survived

```
# testing the model using confusion matrix
predictions = cross_val_predict(random_forest, X_train, Y_train)
confusion_matrix(Y_train, predictions)
```

```
array([[475,  74],
       [ 91, 251]])
```

Mean Score: 81.5%

```
predictions = cross_val_predict(decision_tree, X_train, Y_train)
confusion_matrix(Y_train, predictions)
```

```
array([[449, 100],
       [102, 240]])
```

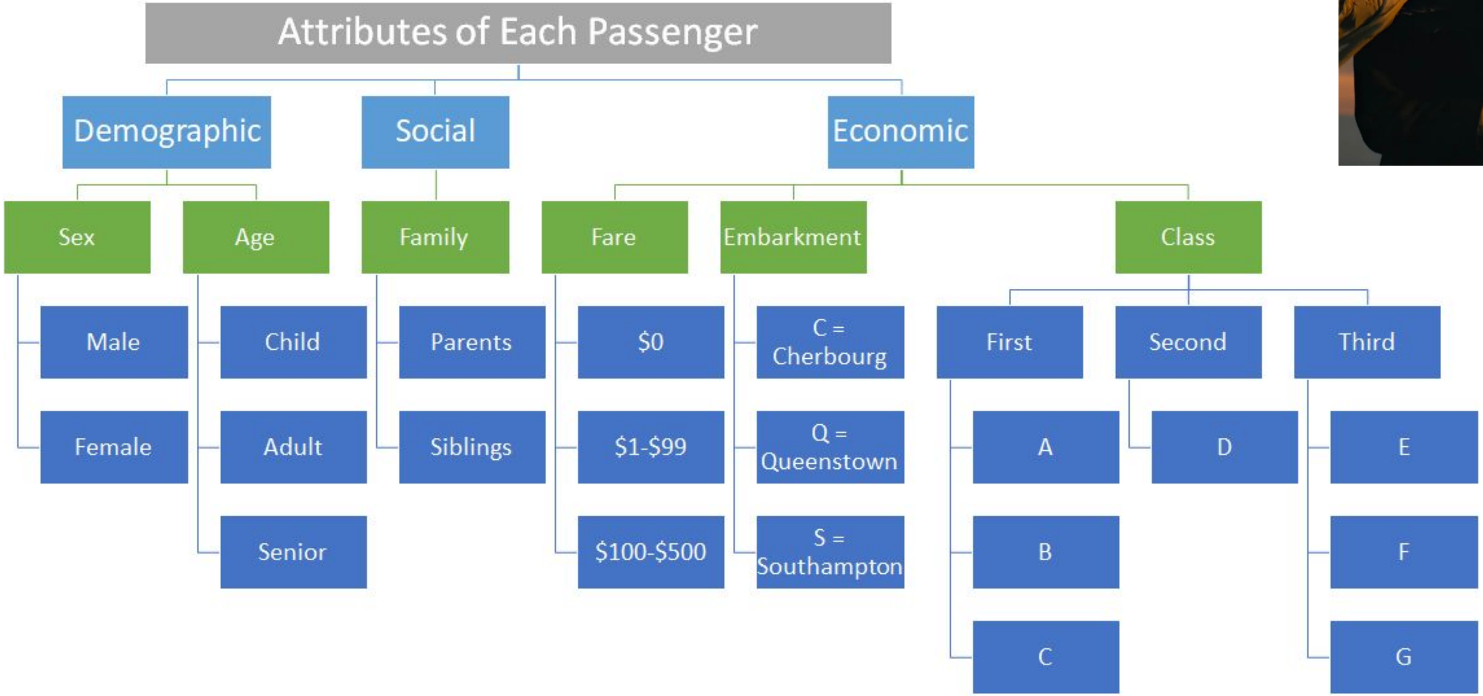
Mean Score: 77.3%

```
predictions = cross_val_predict(logreg, X_train, Y_train)
confusion_matrix(Y_train, predictions)
```

```
array([[463,  86],
       [106, 236]])
```

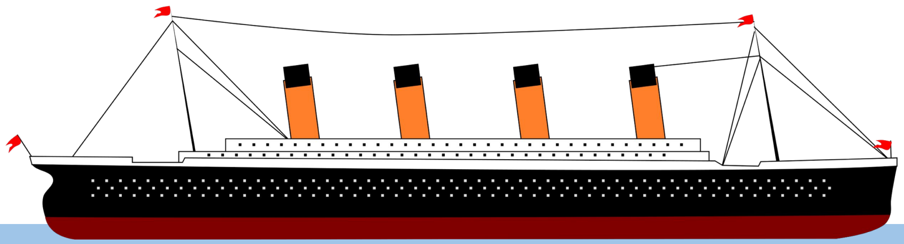
Mean Score: 78.8%

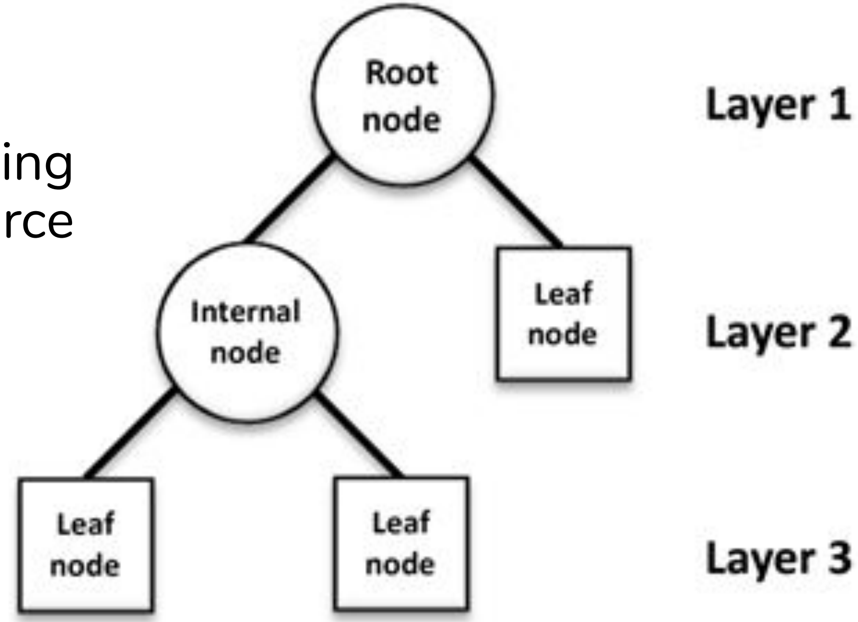
Passenger Data Analyzing



ML models applied - Decision Tree

- **Decision tree** uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.
- Keyword: **Branch**

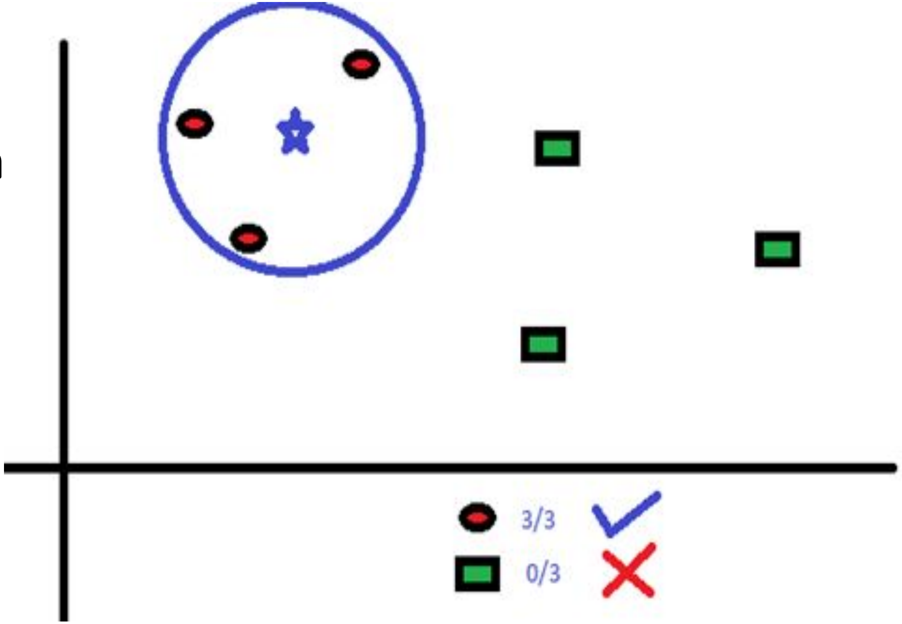




https://www.researchgate.net/figure/Basic-structure-of-a-decision-tree-All-decision-trees-are-built-through-recursion_fig3_295860754

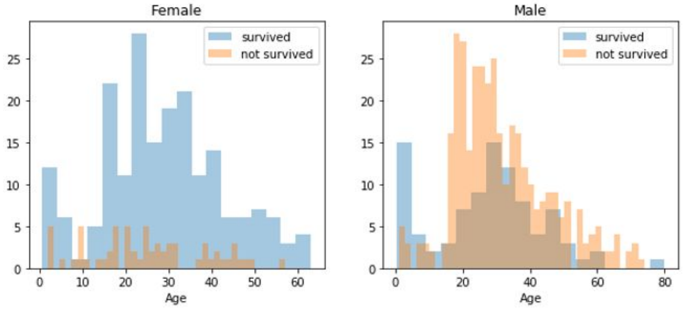
ML models applied - K Nearest Neighbor

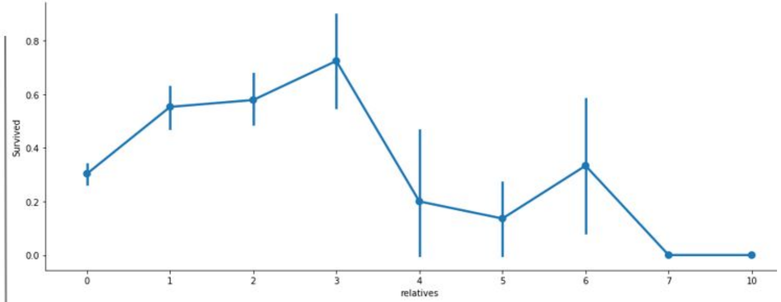
- **K nearest neighbors** is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure.
- Keyword: **Cluster**

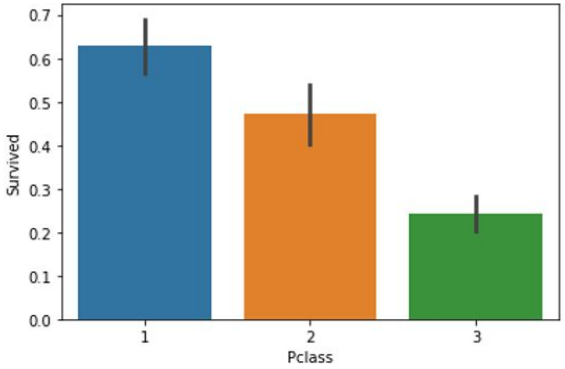


https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

The process of looking for relevant features







	Total	%
Cabin	687	77.1
Age	177	19.9
Embarked	2	0.2
Fare	0	0.0
Ticket	0	0.0

Score and fun discovery

	Model	Score
0	Decision Tree	92.70
1	K Nearest Neighbour	86.76

```
for dataset in data:
    mean = train_df["Age"].mean()
    std = test_df["Age"].std()
    is_null = dataset["Age"].isnull().sum()
    # compute random numbers between the mean, std and is_null
    rand_age = np.random.randint(mean - std, mean + std, size = is_null)
    # fill NaN values in Age column with random values generated
    age_slice = dataset["Age"].copy()
    age_slice[np.isnan(age_slice)] = rand_age
```

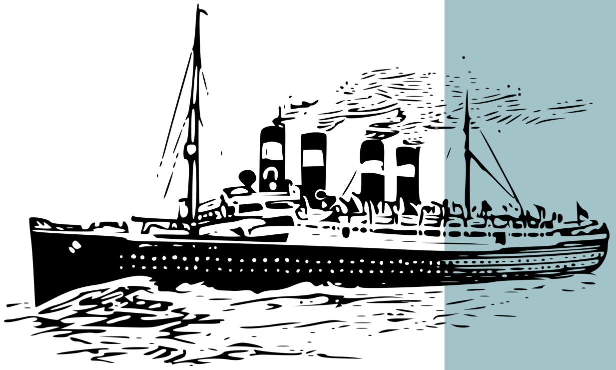
Data Preprocessing in Pandas

```
<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
```

```
embark=pd.get_dummies(titanic_data["Embarked"],drop_first=True)
embark_test=pd.get_dummies(test_data["Embarked"],drop_first=True)
embark.head()
```

	Q	S
0	0	1
1	0	0
2	0	1
3	0	1
4	0	1

Modelling



```
titanic_data=pd.concat([titanic_data,sex,embark,Pcl],axis=1)
titanic_data.head(2)
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	male	Q	S	2	3
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S	1	0	1	0	1

```
test_data.drop(["Sex","Embarked","Name","Ticket","Pclass"],axis=1, inplace=True)
test_data.head(2)
```

	Survived	Age	SibSp	Parch	Fare	male	Q	S	2	3
0	0	22.0	1	0	7.2500	1	0	1	0	1
1	1	38.0	1	0	71.2833	0	0	0	0	0
2	1	26.0	0	0	7.9250	0	0	1	0	1
3	1	35.0	1	0	53.1000	0	0	1	0	0
4	0	35.0	0	0	8.0500	1	0	1	0	1

Score	Model
80.99	Random Forest
79.42	Naive Bayes
78.29	Decision Tree
70.75	Support Vector Machines
70.31	KNN

Project Final Conclusions

1. How someone manipulates data can produce different machine learning outcomes/accuracy scores
    a. Dropping null values decreases ML accuracy
    b. ML does not like non-numeric values either
    c. A “perfect” ratio of features/variables - can have too many or not enough
2. Different ML algorithms can produce vastly different predictions and accuracy scores even when running through the same data
3. You can guess fairly accurate predictions and make fair hypotheses with Pandas (and other visualization tools) on initial data, even before applying ML
4. The more accurate and robust the dataset, the better ML can “learn” from the data to make better, more sound predictions

Thank you, questions?