**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Titanic Walk Through**

**CAC 350**

As an example of how we might run through a problem, we will analyze the Titanic dataset. Some of this content is from Titanic Data Science Solutions[[1]](#footnote-1) on Kaggle.

There are eight steps to any Machine Learning project:

1. Frame the problem and look at the big picture
2. Get the data
3. Explore the data to gain insights
4. Prepare the data to better expose the underlying data patterns to Machine Learning algorithms
5. Explore many different models and shortlist the best ones
6. Fine-tune your models and combine them into a great solution
7. Present your solution
8. Launch, monitor, and maintain your system

Please complete a Machine Learning system for analyzing the Titanic data. Some parts we will do together and some parts, I will ask you to work with a partner.

The following questions are to guide you in completing a Machine Learning system.

Frame the Problem:

1. Why is it important to frame the problem before looking at the data?
2. The problem as stated by Kaggle due to this being a competition problem:

The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered “unsinkable” RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren’t enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

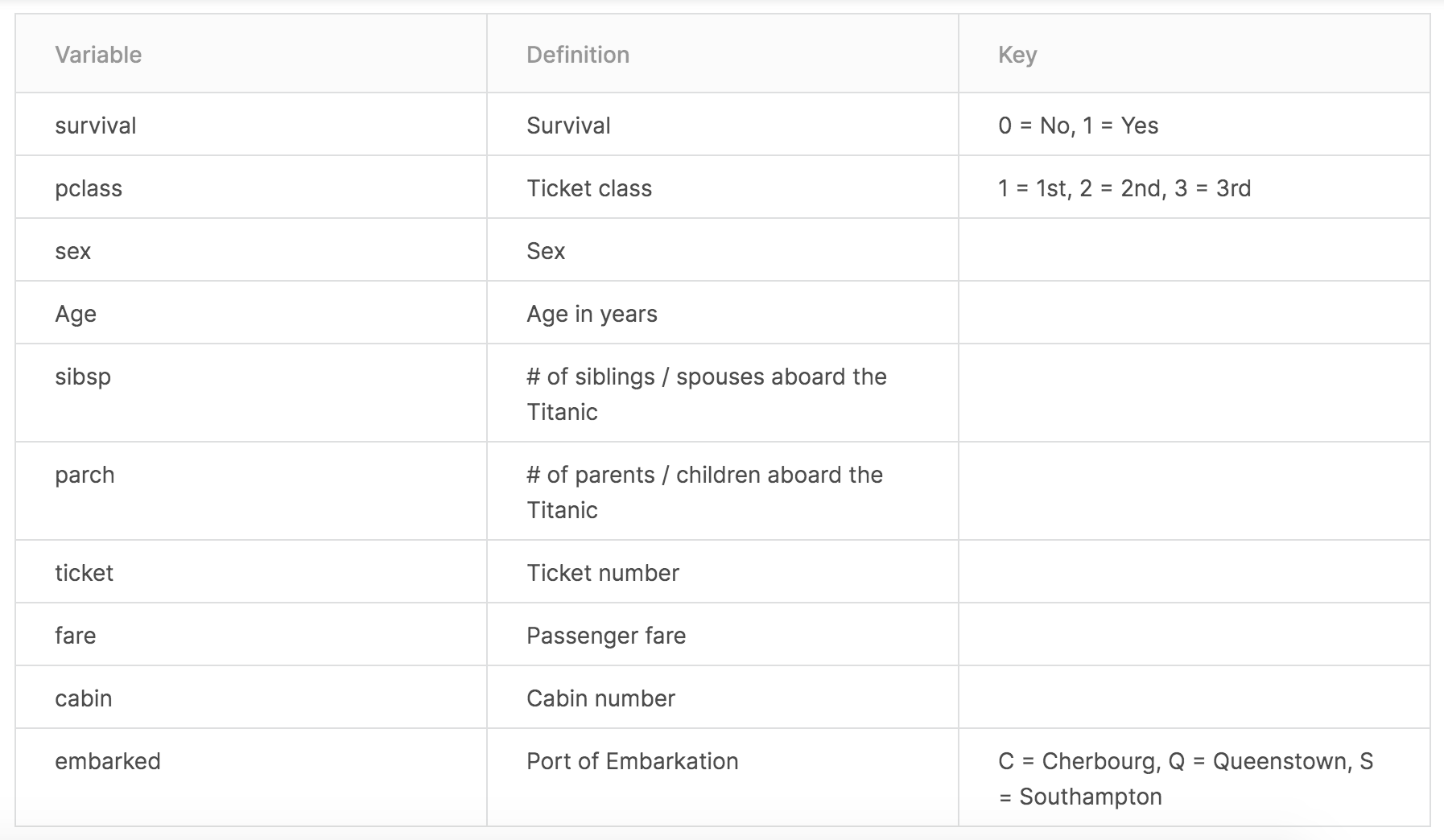
In this challenge, we ask you to build a predictive model that answers the question: “what sorts of people were more likely to survive?” using passenger data (i.e., name, age, gender, socio-economic class, etc).

Question: what type of problem is this?

Get the Data:

1. You can download it from Moodle. It is already broken up into training and test.

Explore the Data:   
Here is some information about the data provided by Kaggle:



pclass: A proxy for socio-economic status (SES)

1st = Upper

2nd = Middle

3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...

Sibling = brother, sister, stepbrother, stepsister

Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way...

Parent = mother, father

Child = daughter, son, stepdaughter, stepson

Some children travelled only with a nanny, therefore parch=0 for them.

1. What are the column headings?

Knowing the type of data we’re dealing with can help us choose appropriate visualizations as well as know how we might need to manipulate (transform) the data.

1. Which features are categorical, and of those that are categorical, what type of category are they (nominal, ordinal, ratio, or interval based)?

Ordinal: PassengerID (may not be relevant), Ticket (may not be relevant), Pclass

Nominal: Embarked, Sex, Cabin, Survived

1. Which features are numerical? Are they discrete, continuous, or timeseries based?

Discrete: SibSp, Parch

Continuous: Age, Fare

1. Preview the first five rows of the data. Preview the last five rows of the data. What do you observe about the data?
2. Are any features missing data? Are there null values? If so, which features?
3. Describe the data both numerical and string (hint: describe(include=[‘0’]) allows strings to be described statistically[[2]](#footnote-2)).

Next steps…based on what we have learned so far, we may want to know how well each feature correlates with Survival. We need to clean some data. This means, we may want to complete some missing values if we feel the feature is important, correct others, which may mean dropping the feature entirely. We may want to create new features to simplify the data.

1. By pivoting features against each other, we can quickly measure correlation. Example: train\_df[[‘Sex’,’Survived’]].groupby([‘Sex’], as\_index=False).mean().sort\_values(by=’Survived’,ascending=False)

Grab a partner and determine which features you should try and explain the

results.

1. Try a histogram to look at age. A histogram is useful for analyzing continuous numerical variables where banding or ranges will help identify useful patterns.

g = sns.FacetGrid(train\_df, col='Survived')

g.map(plt.hist, 'Age', bins=20)

What do the results demonstrate?

1. Do the same as #11 but use Pclass and Age. You’ll need to add a row to the FacetGrid for Pclass. Suggested: add size = 2.2 and aspect = 1.6. In the grid.map add alpha =.5. Add a legend to the grid.

What do the results demonstrate?

1. What about categorical features? We can use a point plot to look at survival by class and sex. Point plots are good for focusing comparisons between different levels of one or more categorical variables. They are very good for looking at interactions. For example, they can show how the relationship between levels of one categorical variable changes across levels of a second categorical value.

grid = sns.FacetGrid(train\_df, row='Embarked', size=2.2, aspect=1.6)

grid.map(sns.pointplot, 'Pclass', 'Survived', 'Sex', palette='deep')

grid.add\_legend()

What do the results demonstrate?

1. Let’s take a look at fare and see how it correlates with embarked, sex, and survival.

grid = sns.FacetGrid(train\_df, row='Embarked', col='Survived', size=2.2, aspect=1.6)

grid.map(sns.barplot, 'Sex', 'Fare', alpha=.5, ci=None)

grid.add\_legend()

Prepare the Data:

1. We likely need to drop a few features…What are two that you think we should drop?
2. Can we salvage anything from Name? Replace rare names with Rare, Mlle, Ms to Miss, and Mme to Mrs. Then do a groupby to see what the results look like.
3. Convert nominal titles to ordinal.
4. Drop Name now that we snagged title…anything else we should drop?
5. For many models, we need numerical values rather than strings so let’s adjust the Sex from strings to numbers.
6. Feature completion. When we have missing values, it can cause problems with models. If we’re not missing too many, we should try to complete them. Let’s look at Age and see how it relates to Sex and PClass.

Write the code to loop through the datasets, Sex, and PClass to find the

median age for each grouping. Then, replace that information in the dataset.

1. Now, we can create AgeBands. Create a new feature using pd.cut with 5 bands.
2. Now, change Age into ordinals based on the band. Find the location where Age falls within a certain range and replace that with 0, 1, 2, etc. based on the band.
3. Drop AgeBand since we have it stored in Age now.
4. Just to experiment, try creating a new feature called FamilySize that sums SibSp and Parch.
5. Create a new feature called IsAlone that equals 1 if FamilySize is 1; otherwise, it equals 0.
6. Which new feature is more valuable? Should you consider dropping any?
7. Complete the categorical variable of embarked by filling in the empty values with the most common port (mode). Then, convert them to numeric values.
8. For a numeric feature, we can complete it using the median value. For Fare, replace any missing values with the median. Try creating FareBands (4) and replacing Fare with the band number.
9. Time to try some modeling…

1. https://www.kaggle.com/startupsci/titanic-data-science-solutions [↑](#footnote-ref-1)
2. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.describe.html [↑](#footnote-ref-2)