Lesson #21
Kaggle Fundamentals

notas

Getting Started with Kaggle

Kaggle Workflow

- Load Libraries
- Get data, including EDA
- Clean, prepare and manipulate Data (feature engineering)
- Modeling (train and test)
- Algorithm Tuning
- Finalizing the Model (submission)

notas

kaggle

What is Kaggle?
Why I Participate?
What is the Impact?

- Competitions
- Datasets
- Notebooks
- Discussion
- Courses
- Jobs
- Social Network
- ...

Getting Started Prediction Competition

Titanic: Machine Learning from Disaster

Start here! Predict survival on the Titanic and get familiar with ML basics

Kaggle · 11,209 teams · Ongoing

Overview

👋 🚢 Ahoy, welcome to Kaggle! You're in the right place.

This is the legendary Titanic ML competition – the best, first challenge for you to dive into ML competitions and familiarize yourself with how the Kaggle platform works.

The competition is simple: use machine learning to create a model that predicts which passengers survived the Titanic shipwreck.

Read on or watch the video below to explore more details. Once you're ready to start competing, click on the "Join Competition button to create an account and gain access to the competition data. Then check out Alexis Cook's Titanic Tutorial that walks you through step by step how to make your first submission!

Getting Started Prediction Competition

Titanic: Machine Learning from Disaster

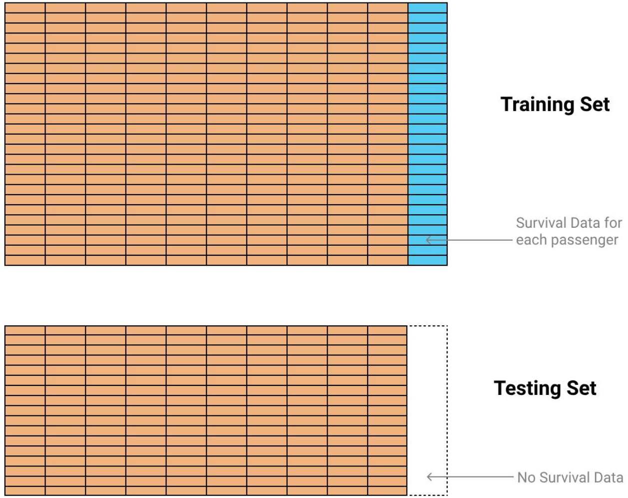
Start here! Predict survival on the Titanic and get familiar with ML basics

Kaggle · 11,209 teams · Ongoing

Overview | Data | Notebooks | Discussion | Leaderboard | Rules | Team | My Submissions | Submit Predictions

Data Sources

gender_submission.csv	418 x 2
test.csv	418 x 11
train.csv	891 x 12



Overview Data Notebooks Discussion Leaderboard Rules Team My Submissions **Submit Predictions**

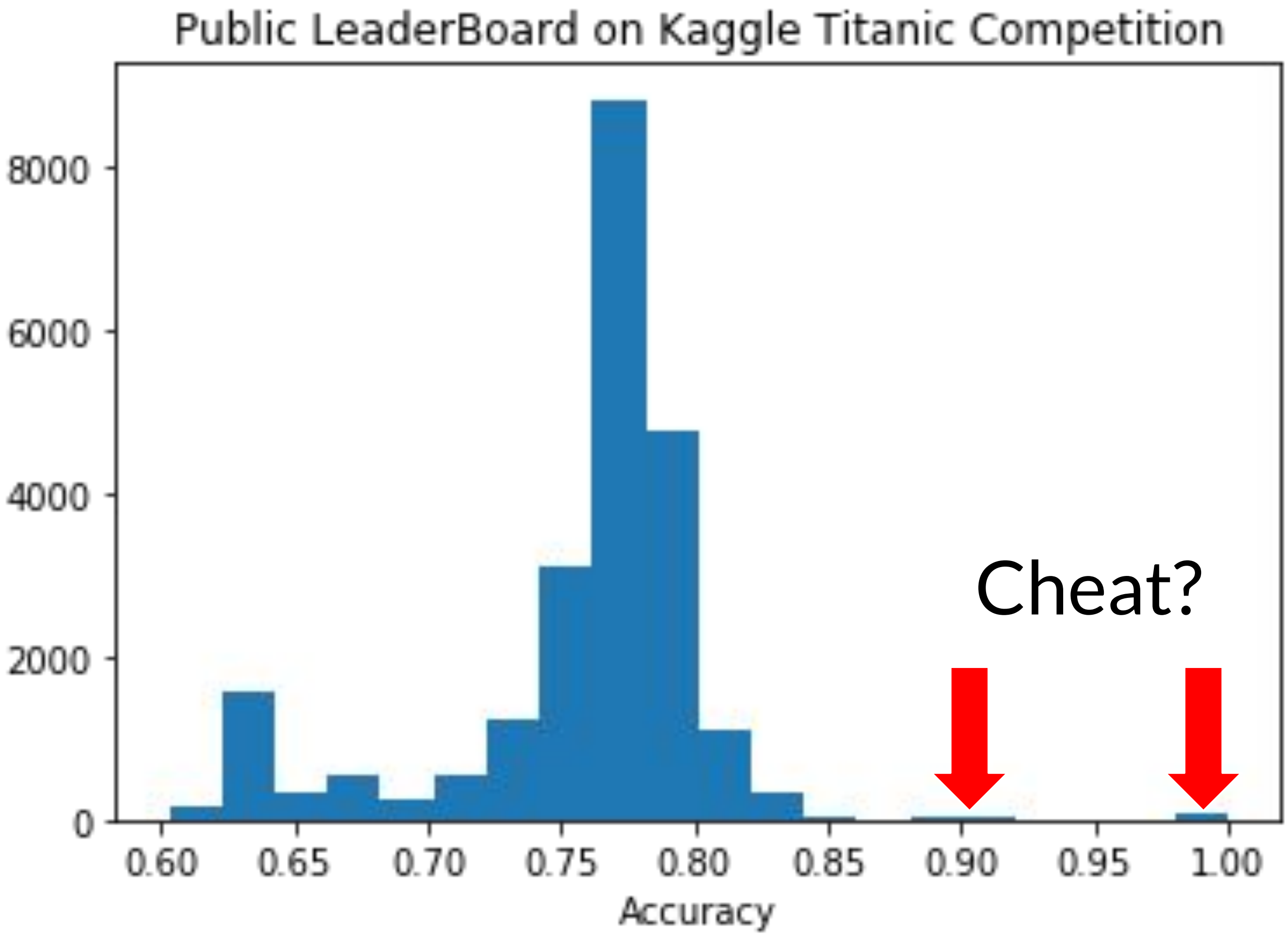
Public Leaderboard **Private Leaderboard**

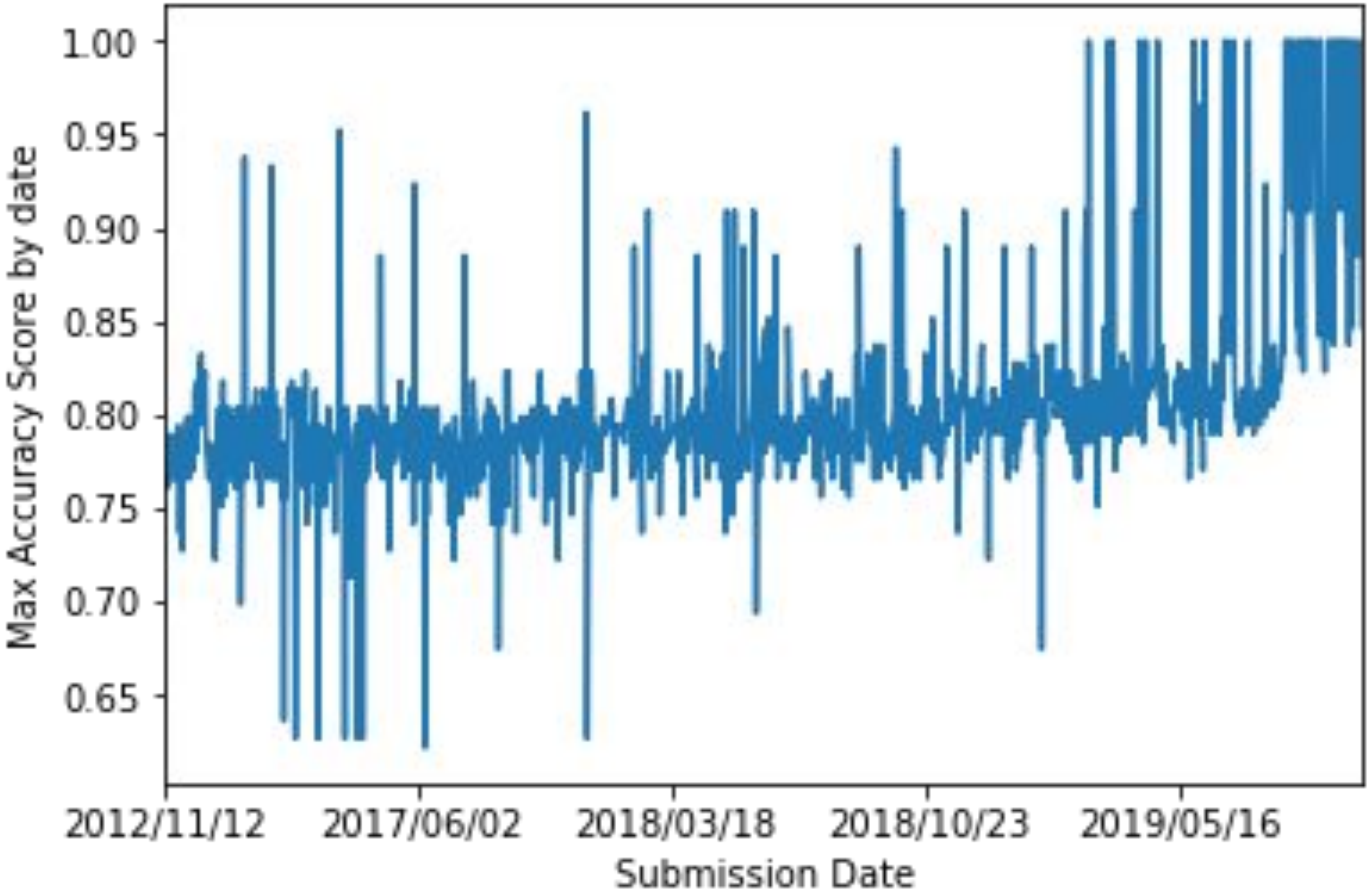
This leaderboard is calculated with approximately 50% of the test data.

The final results will be based on the other 50%, so the final standings may be different.

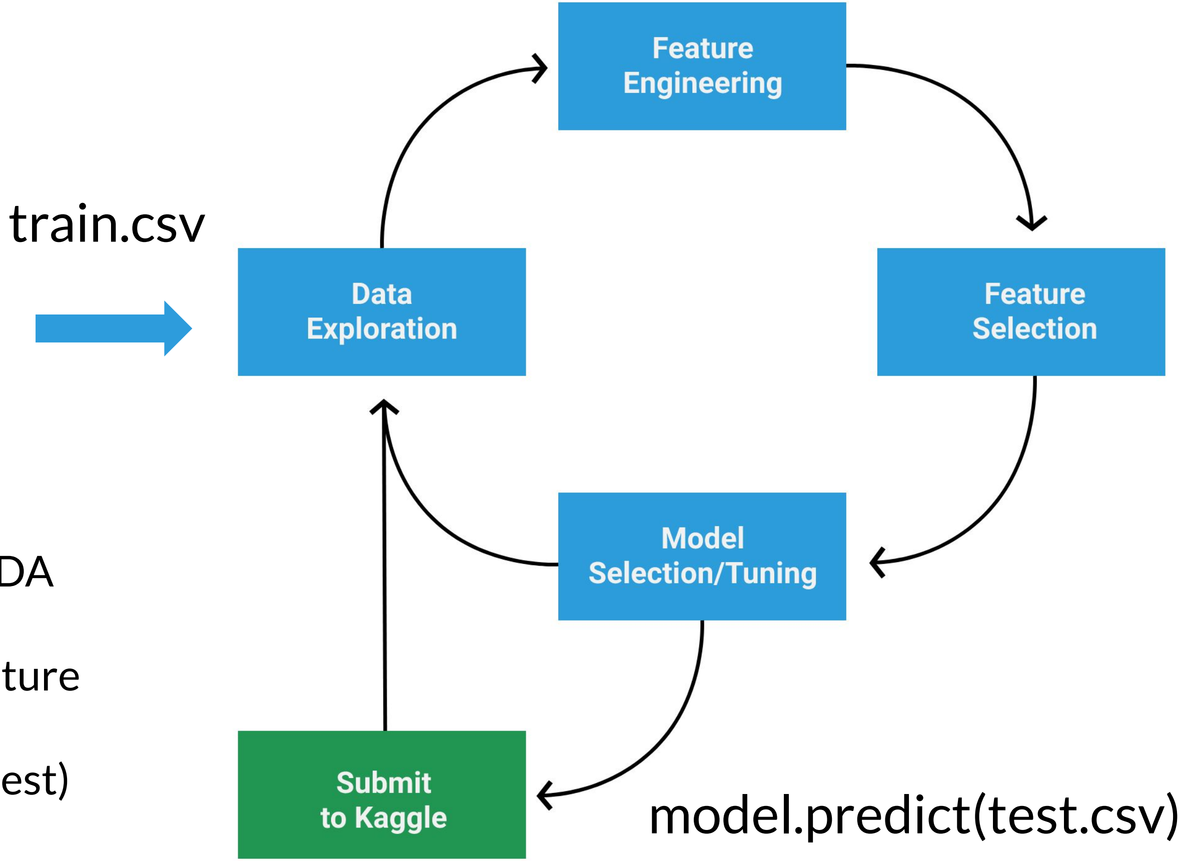
Raw Data Refresh

#	Team Name	Notebook	Team Members	Score	Entries	Last
1	**Reza R Pratama**			1.00000	1	2mo
2	**Matheus Silva**			1.00000	1	2mo
3	**Batsy**			1.00000	1	2mo
4	**Patrick Bruecker**			1.00000	1	2mo
5	**SoiSoCiu**			1.00000	25	2mo
6	**ambition12**			1.00000	2	2mo
7	**harshitsheoran**			1.00000	1	2mo
8	**James Strong**			1.00000	1	2mo
9	**chauncey**			1.00000	14	1mo


Public LeaderBoard on Kaggle Titanic Competition
8000
6000
4000
2000
0
0.60
0.65
0.70
0.75
0.80
0.85
0.90
0.95
1.00
Accuracy
Cheat?


Max Accuracy Score by date
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
2012/11/12
2017/06/02
2018/03/18
2018/10/23
2019/05/16
Submission Date

Problem

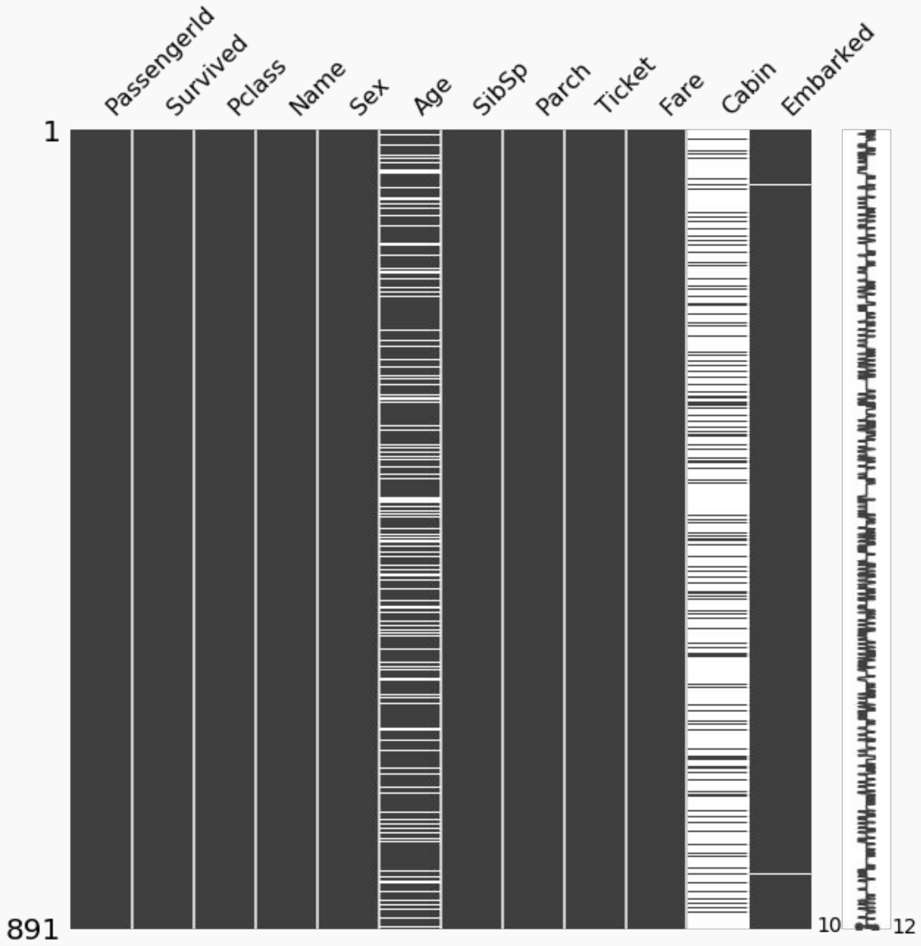


1. Load Libraries
2. Get data, including EDA
3. Clean, prepare and manipulate Data (feature engineering)
4. Modeling (train and test)
5. Algorithm Tuning
6. Creating a submission file

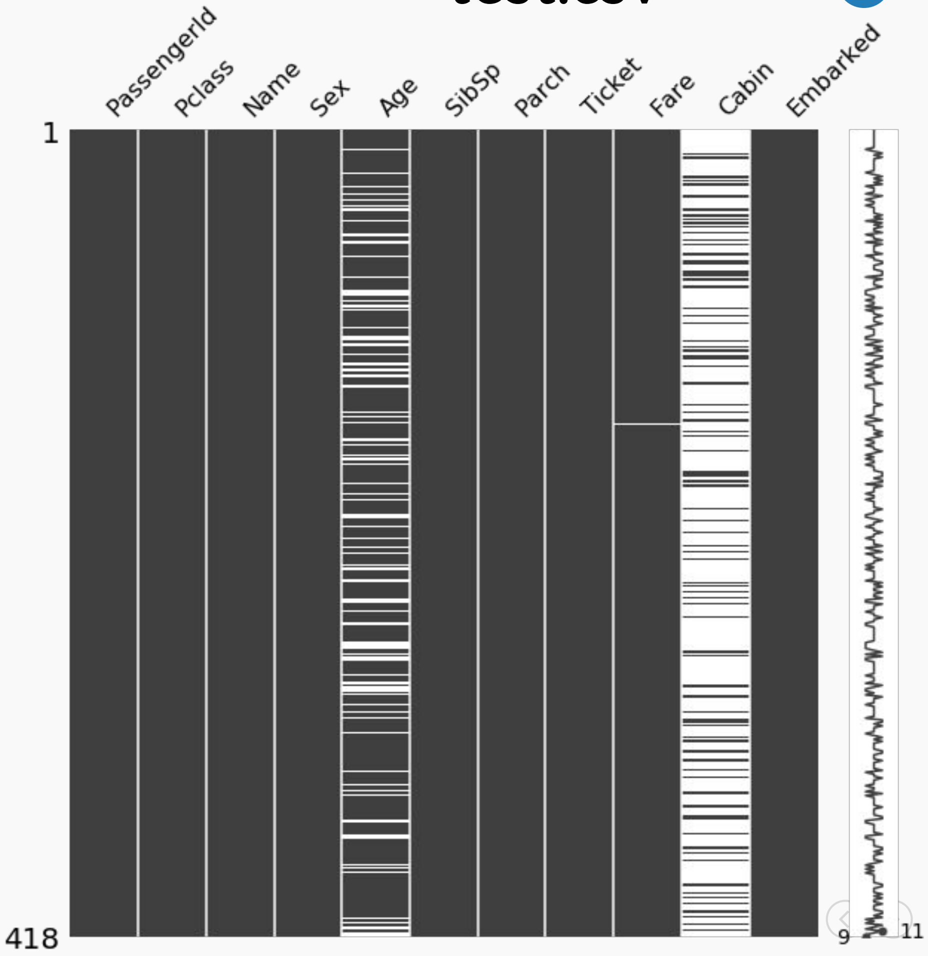
Data Exploration (EDA)

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

train.csv



test.csv

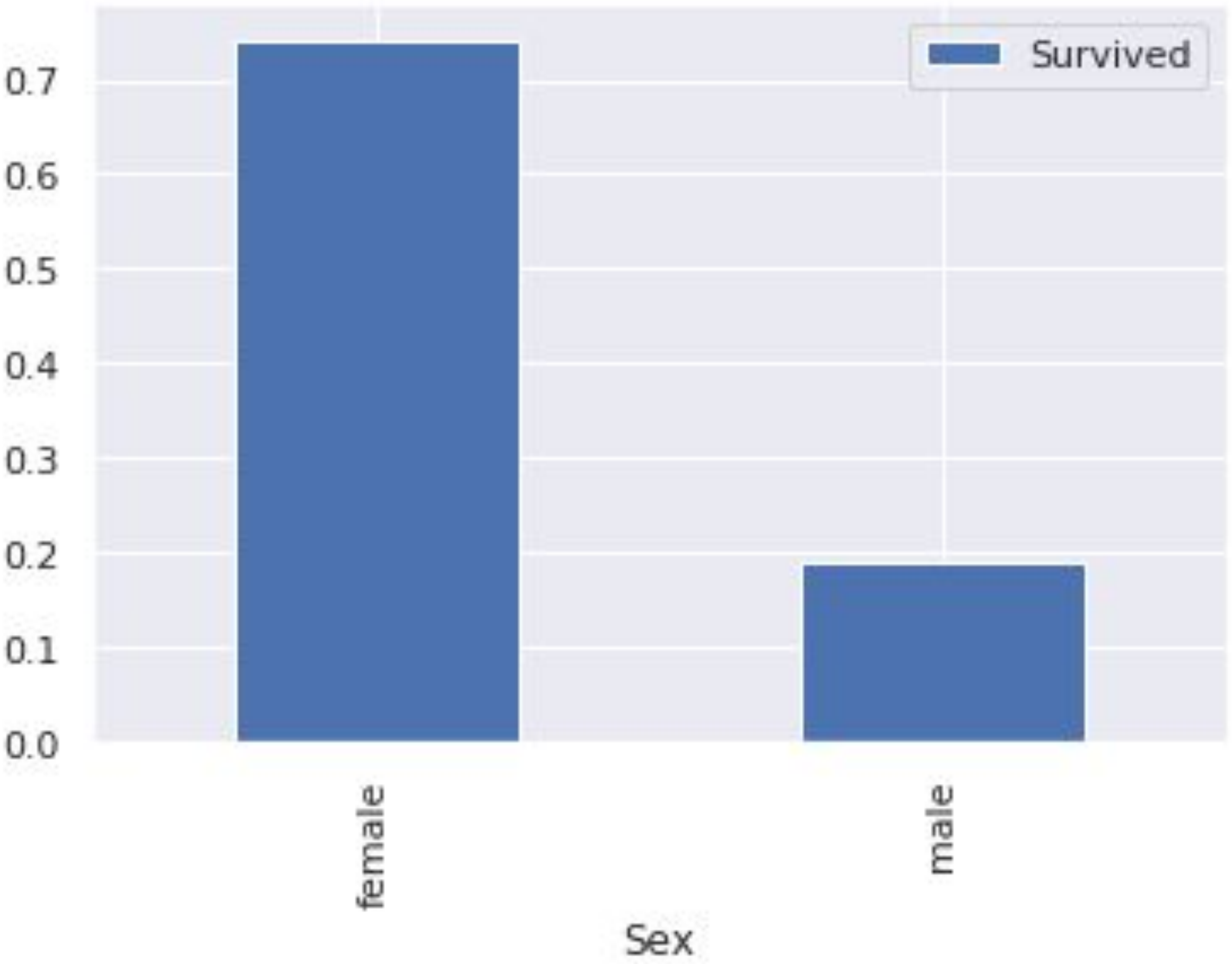


```
train.Survived.value_counts()
```

```
0      549
1      342
```

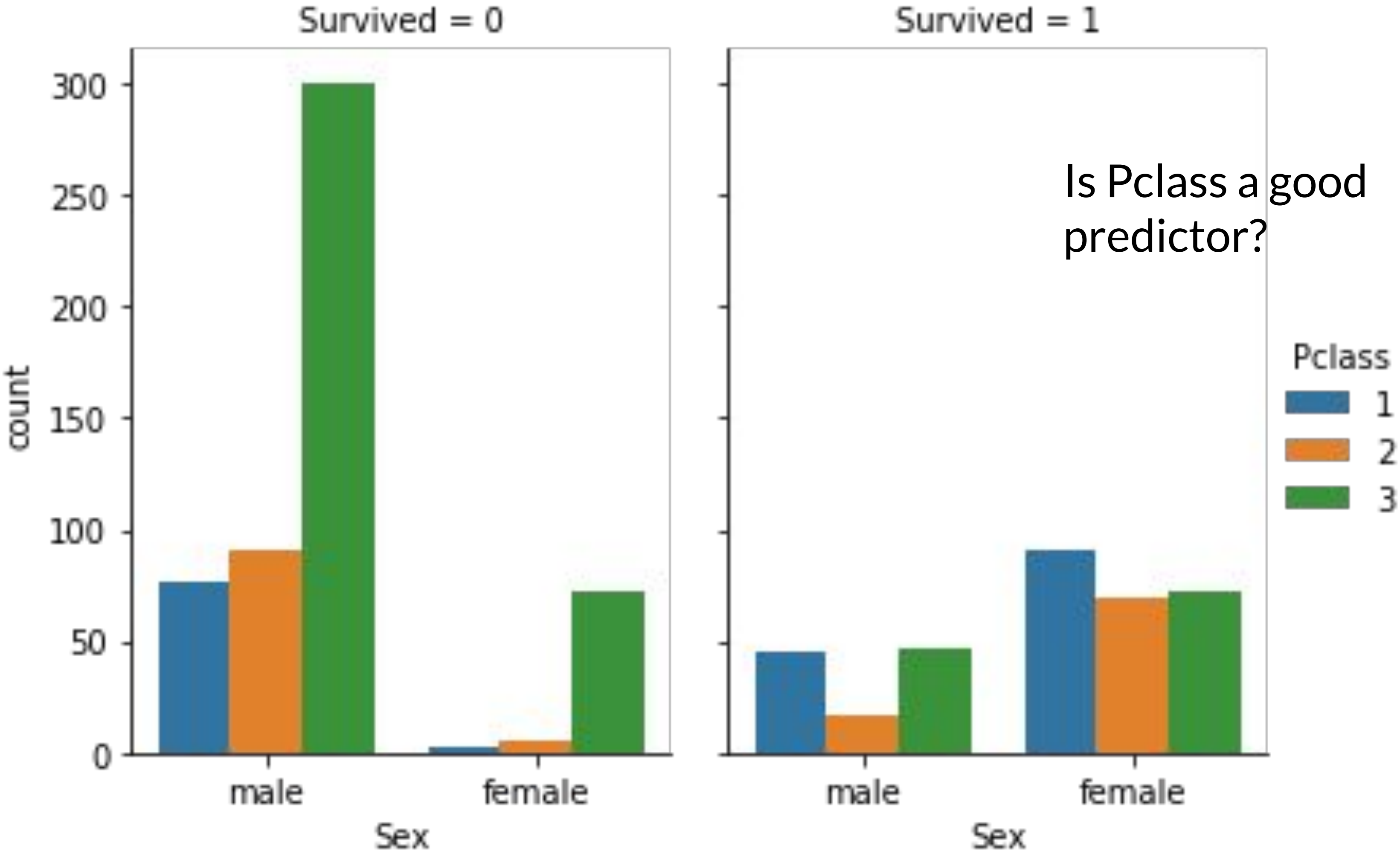
Hypothesis #01 (naive)

The simplest strategy of guessing that all died since the majority died

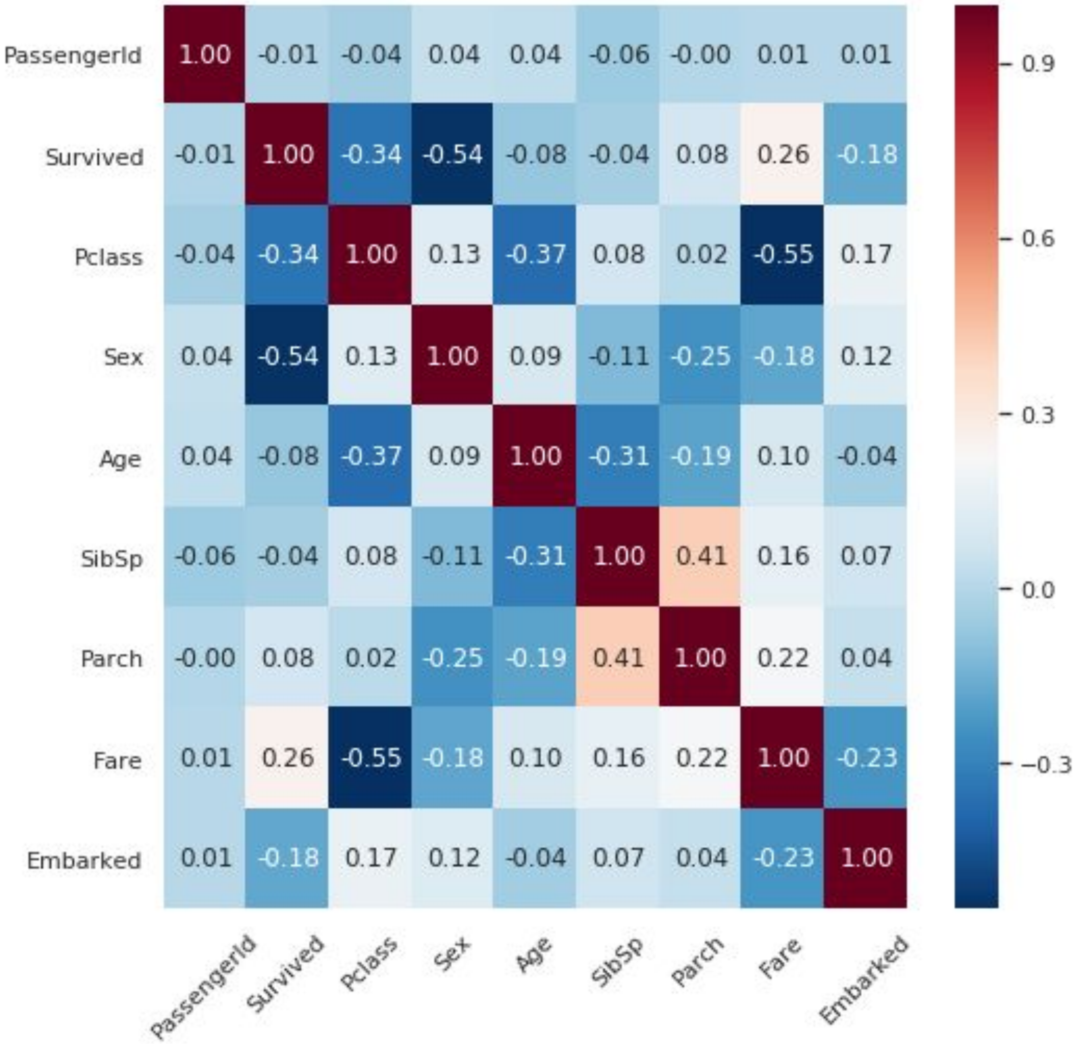

Survived
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
female
male
Sex

Hypothesis #02

Since roughly 75% of females survived and roughly only 20% of males survived, what's the score when you guess all females survived and all males perished?



Is Pclass a good predictor?

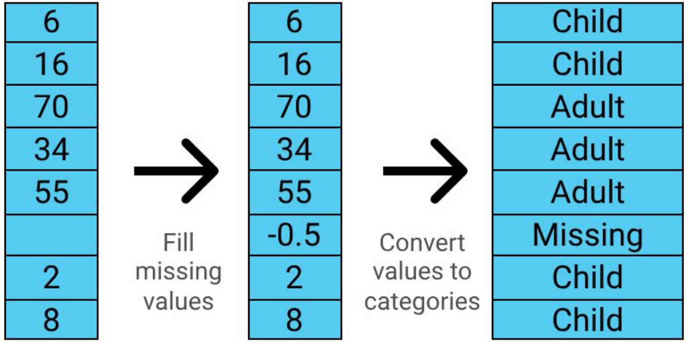


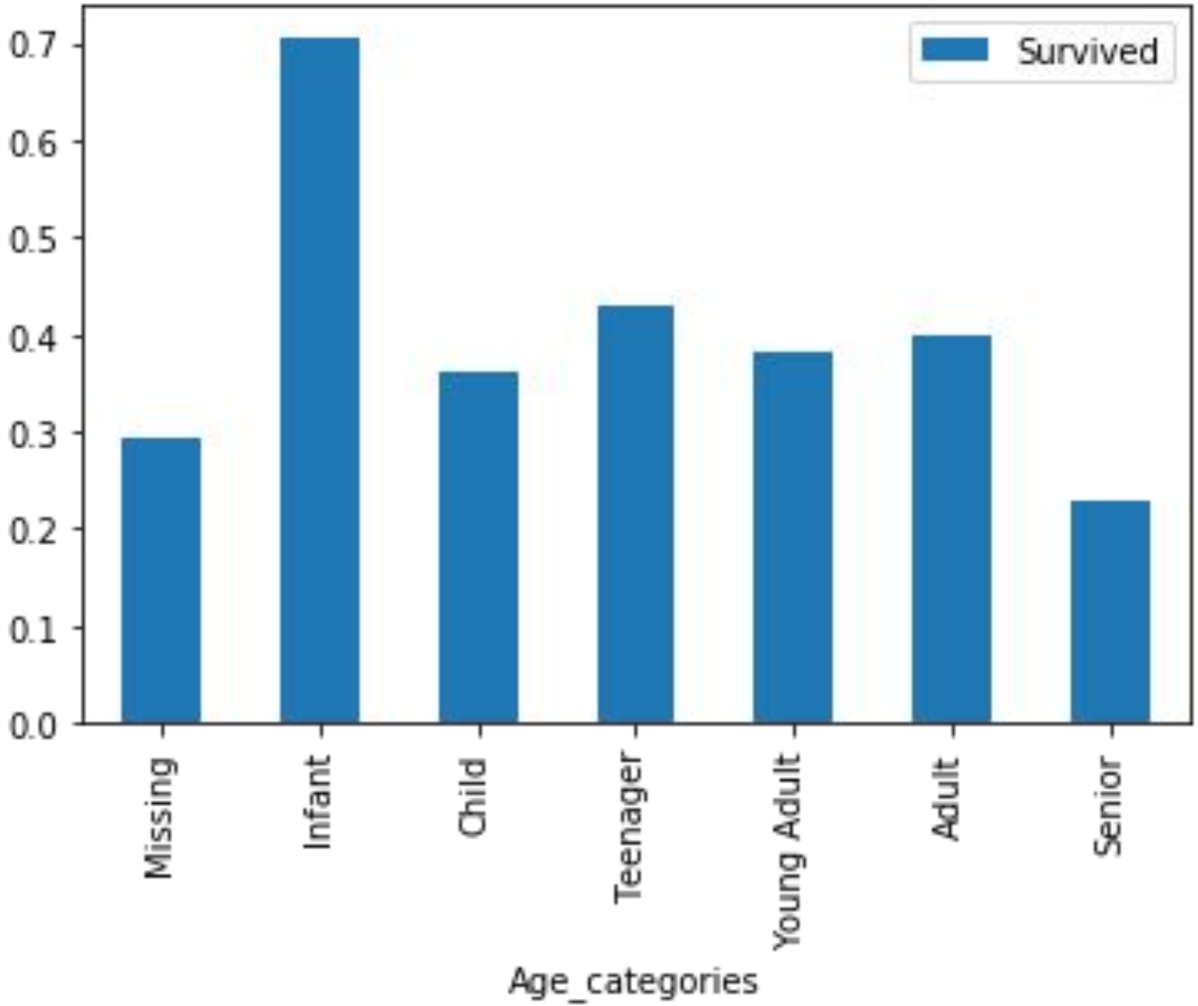
- A correlation of -0.54 shows Sex carries a lot of information about Survived.
- We see then Pclass (-0.34) and Fare (0.26) are the next features that correlate with Survived
- However, Fare and Pclass are very much correlated at (0.55) as we may expect

"Women and Children First"

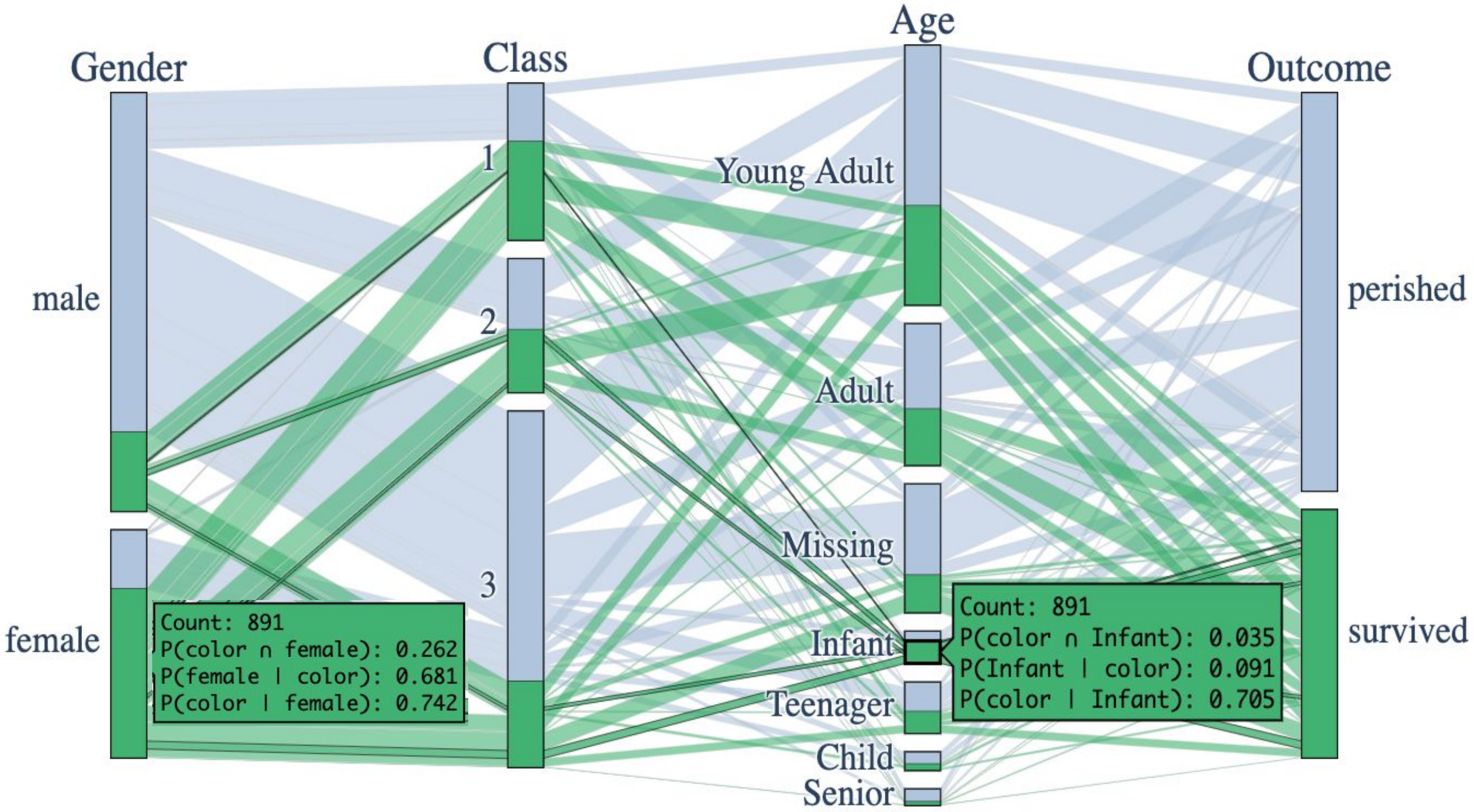
Clean, Preparing and Manipulate Data

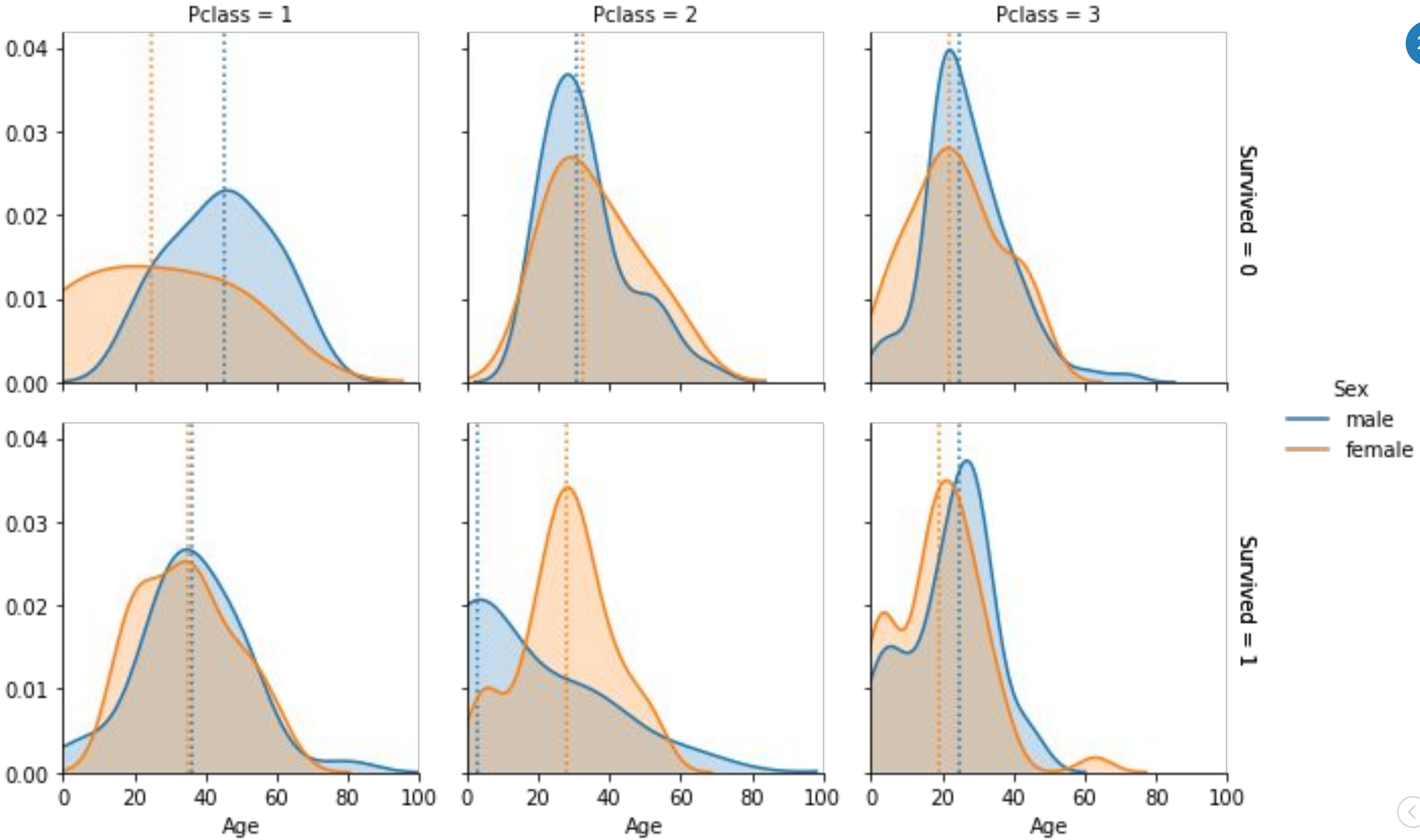
```python
# fill missing values with -0.5
train["Age"] = train["Age"].fillna(-0.5)

# divide age column into a range of values
cut_points = [-1,0,5,12,18,35,60,100]
label_names = ["Missing","Infant","Child",
               "Teenager","Young Adult","Adult","Senior"]
train["Age_categories"] = pd.cut(train["Age"],
                                 cut_points,
                                 labels=label_names)
```





Survived
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Missing
Infant
Child
Teenager
Young Adult
Adult
Senior
Age_categories


Gender
Class
Age
Outcome
male
female
1
2
3
Young Adult
Adult
Missing
Infant
Teenager
Child
Senior
perished
survived
Count: 891
P(color ∩ female): 0.262
P(female | color): 0.681
P(color | female): 0.742
Count: 891
P(color ∩ Infant): 0.035
P(Infant | color): 0.091
P(color | Infant): 0.705



Pclass = 1
Pclass = 2
Pclass = 3
0.04
0.03
0.02
0.01
0.00
Survived = 0
Survived = 1
Sex
male
female
0
20
40
60
80
100
Age

Preparing our Data for Machine Learning

- Sex
- Pclass
- Age_categories

- Before we build our model, we need to prepare these columns for machine learning.
- Most machine learning algorithms can't understand text labels, so we have to convert our values into numbers.

Preparing our Data for Machine Learning

Pclass
3
1
3
1
3
3
1
3
3
2

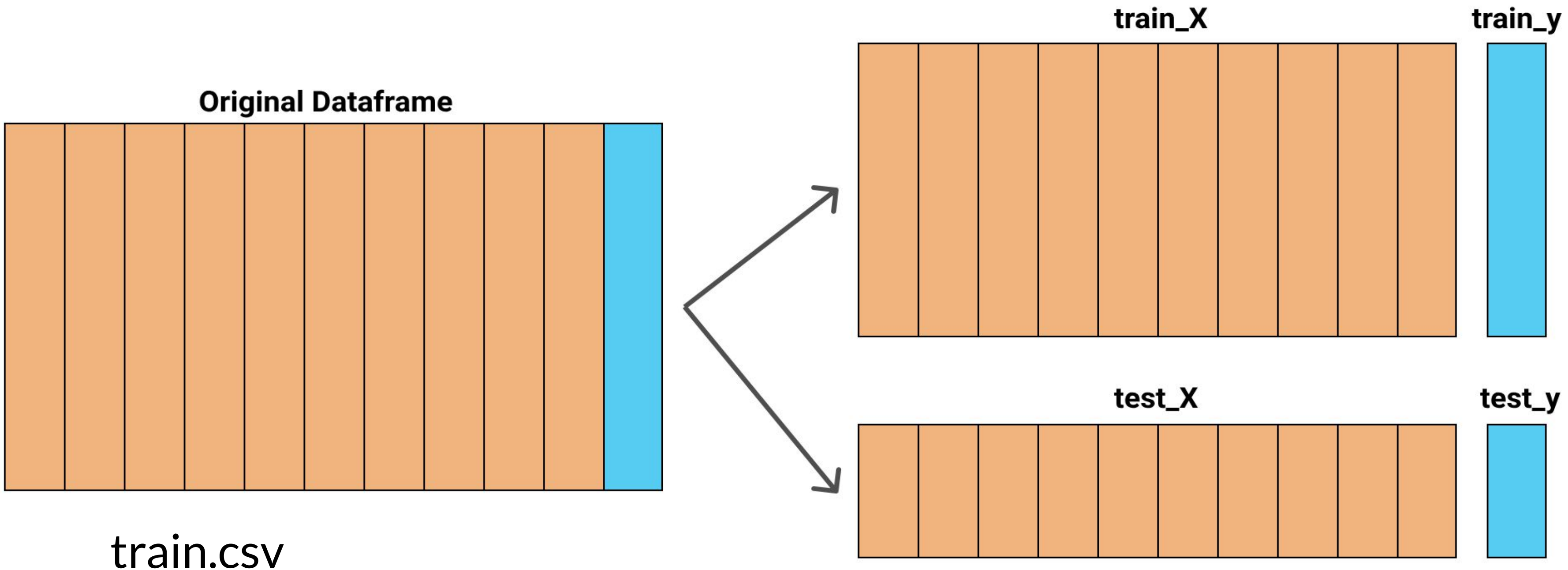
Pclass_1	Pclass_2	Pclass_3
0	0	1
1	0	0
0	0	1
1	0	0
0	0	1
0	0	1
1	0	0
0	0	1
0	0	1
0	1	0

```
def create_dummies(df,column_name):
    # drop_first = True to avoid colinearity
    dummies = pd.get_dummies(df[column_name],
                             prefix=column_name,
                             drop_first=True)
    df = pd.concat([df,dummies],axis=1)
    return df

train = create_dummies(train,"Pclass")
train = create_dummies(train,"Age_categories")
train = create_dummies(train,"Sex")
```

- Model [0]
  - Sex column (categorized), Pclass (raw)
- Model [1]
  - Sex column (get_dummies(drop_first=True)), Pclass (raw)
- Model [2]
  - Sex column (get_dummies(drop_first=True)), Pclass (get_dummies(drop_first=False))
- Model [3]
  - Sex column (get_dummies(drop_first=True)), Pclass (raw), Age (get_dummies(drop_first=False))

Everything ends in **Pipelines**

Creating our First Machine Learning Model



train.csv

PassengerId	Survived
892	0
893	1
894	0

Creating a Submission File

```
predict_final = best_model.best_estimator_.predict(test)

holdout_ids = test["PassengerId"]
submission_df = {"PassengerId": holdout_ids,
                 "Survived": predict_final}
submission = pd.DataFrame(submission_df)

submission.to_csv("submission.csv",index=False)
```

9042	Nikita Nazarov	0.75598	1	17h
9043	tani0	0.75598	2	11h
9044	OTHELLO31	0.75598	2	6h
9045	CHIAKI3	0.75598	6	3h
9046	pallavisonagote	0.75598	5	3h
9047	IvanovitchSilva	0.75598	6	4m

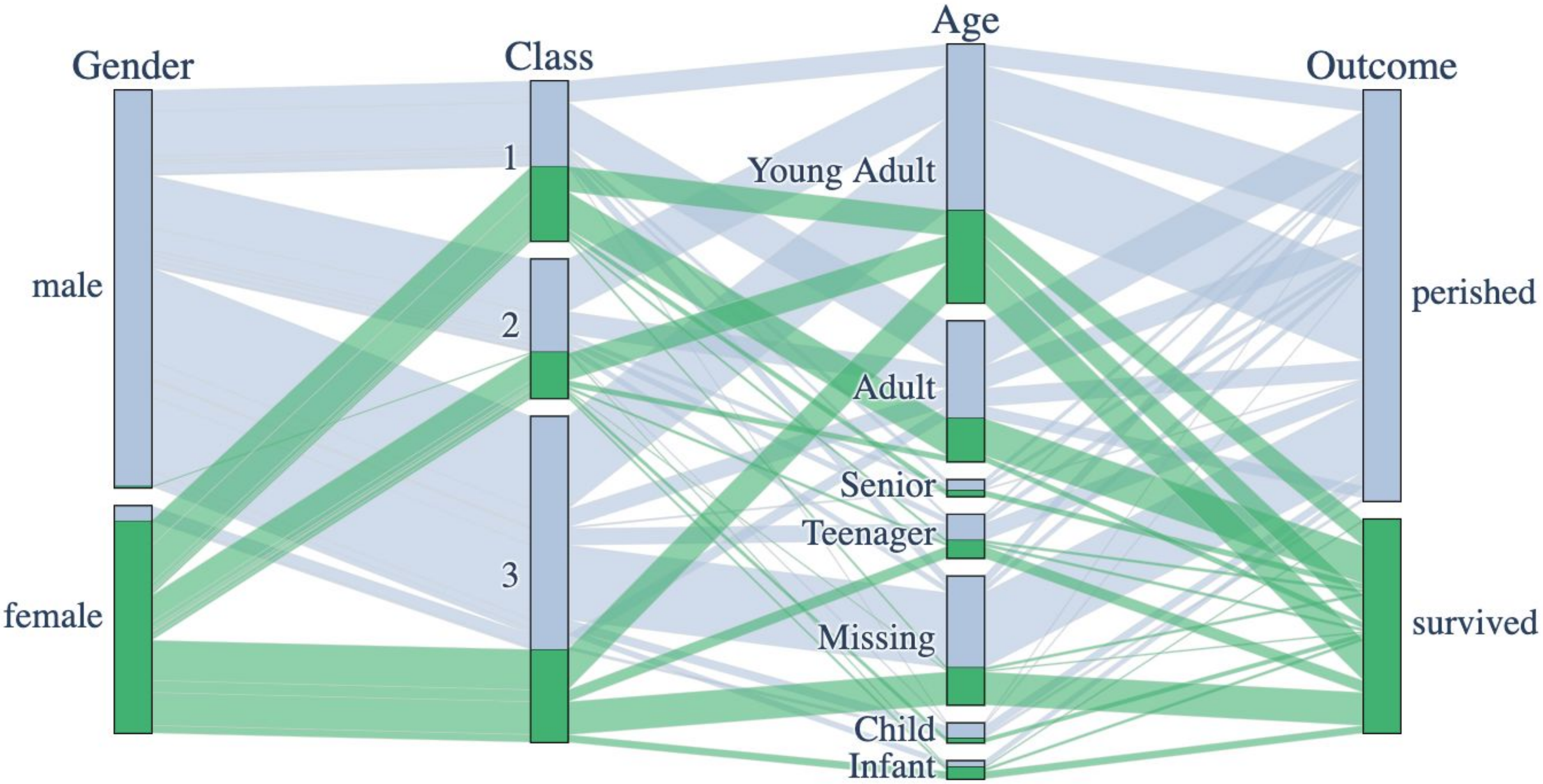
Your Best Entry ↑

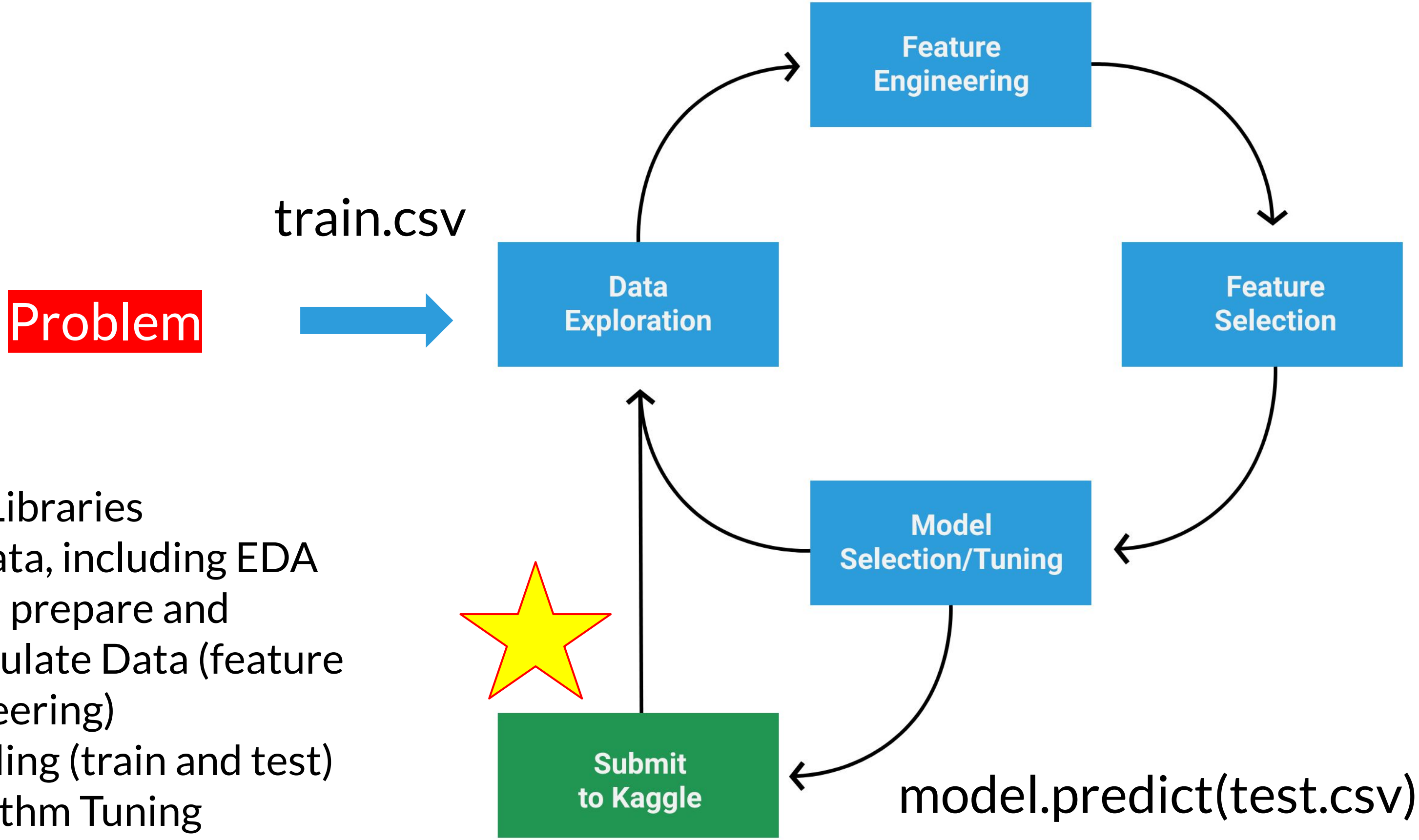
Your submission scored 0.75119, which is not an improvement of your best score. Keep trying!

9048	ctron	0.75119	1	2mo
9049	RyoNamiki	0.75119	1	2mo
9050	nan7674	0.75119	1	2mo
9051	Madhan Varadhodiyil	0.75119	2	2mo
9052	fanxiaohong	0.75119	2	2mo
9053	Kristof Nachtergaele	0.75119	3	2mo
9054	Saurabh_Dalakoti	0.75119	1	2mo
9055	NP_29	0.75119	1	2mo

Model	Features	Parameters	Score	Ratio
All-Dead	-	All zeros	0.62679	-
[0] RandomForest	Sex (categorized) Pclass (raw)	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.74641	19.08%
[1] RandomForest	Sex (dummies(T)) Pclass (raw)	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.74641	19.08%
[2] RandomForest	Sex (dummies(T)) Pclass (dummies(F))	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.74641	19.08%
[3] XGBClassifier	Sex (dummies(T)) Pclass (raw) Age (dummies(F))	learning_rate: 0.001 max_depth: 4 n_estimators: 50	0.75119	19.84%
All-Females Survived All-Males Perished	-	if Sex == "female" Survived = 1 else Survived== 0	0.76555	22.13%

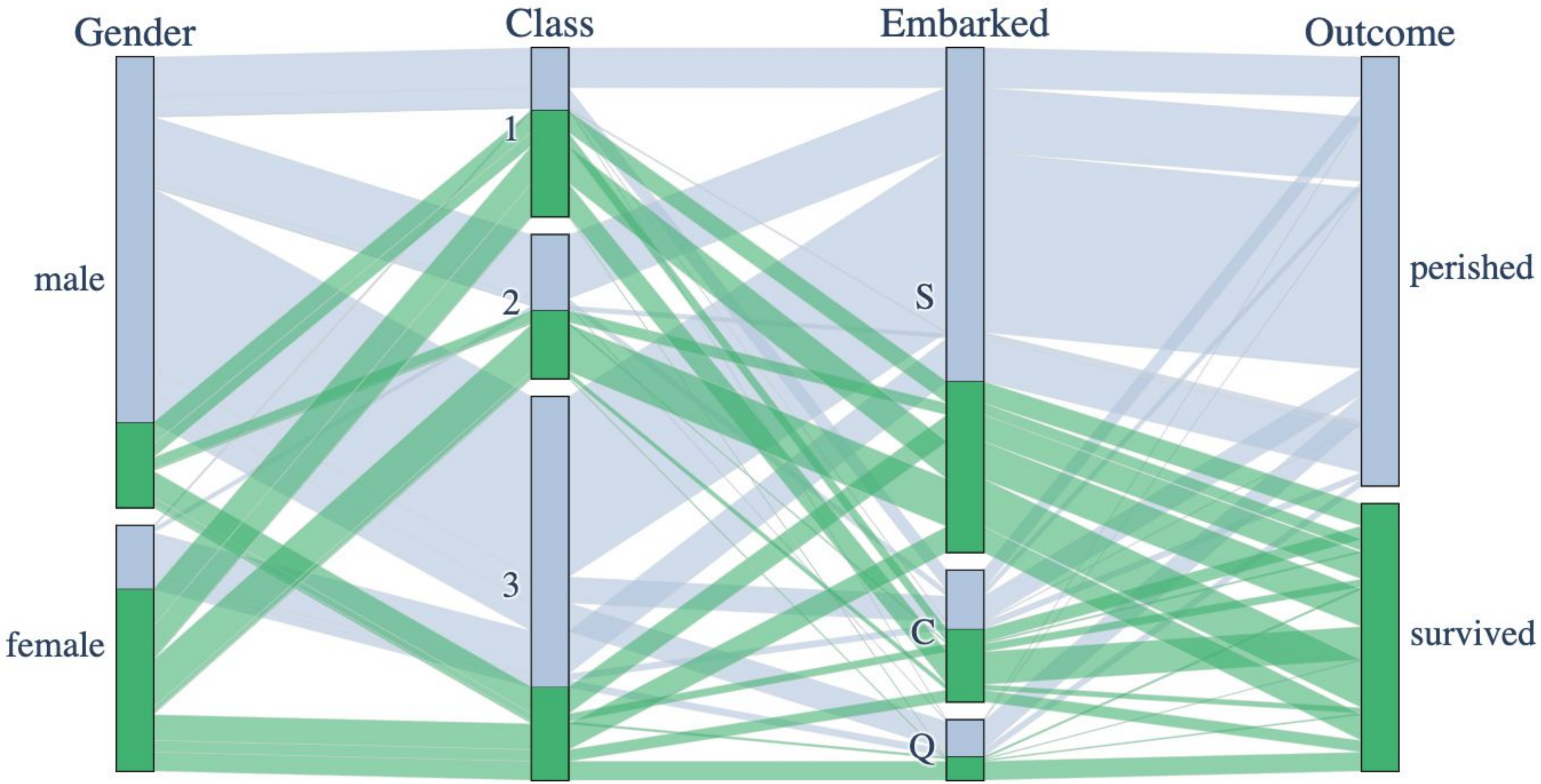
Model #3 - Analyzing the predictions





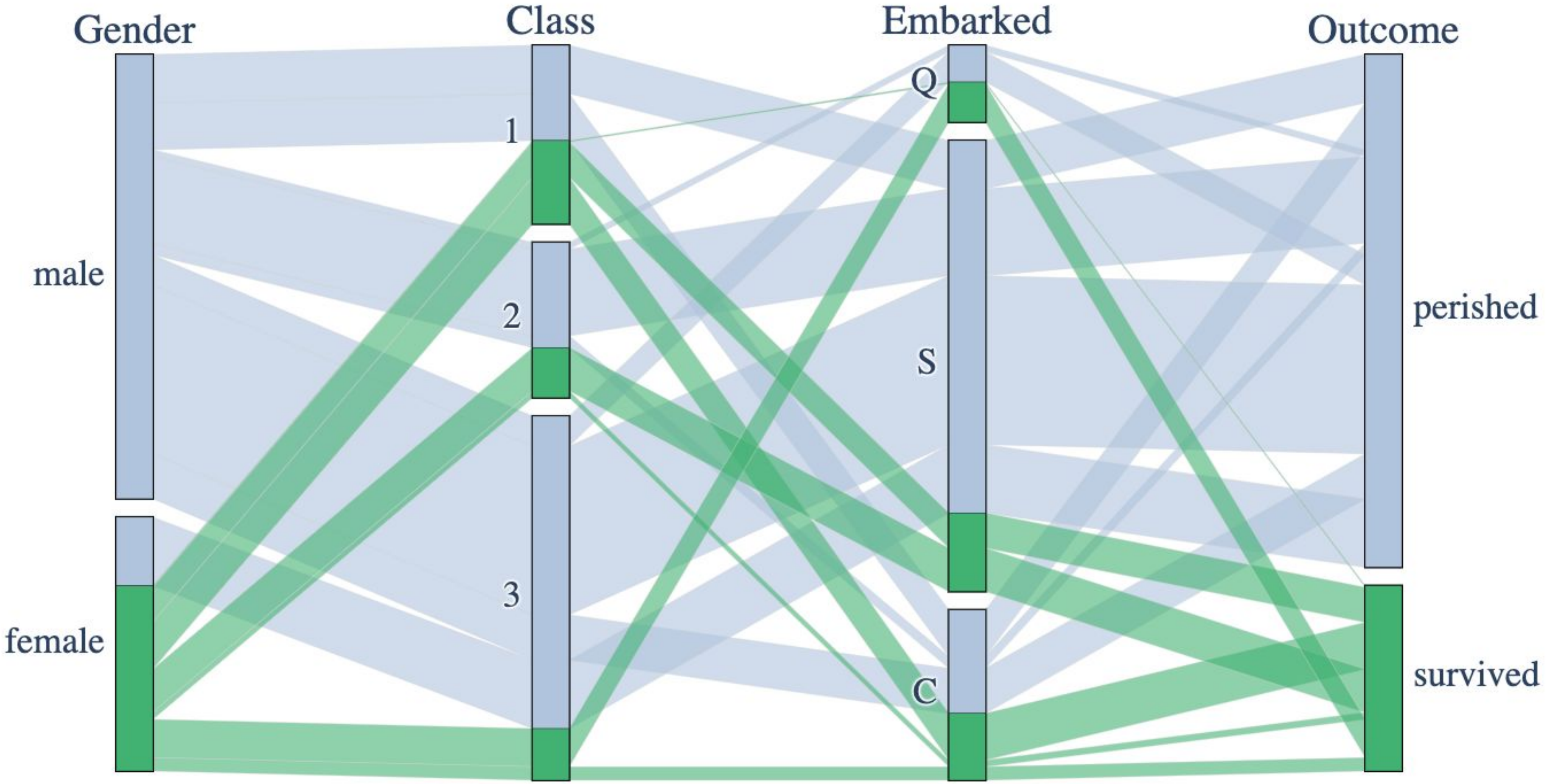
1. Load Libraries
2. Get data, including EDA
3. Clean, prepare and manipulate Data (feature engineering)
4. Modeling (train and test)
5. Algorithm Tuning
6. Creating a submission file

What about adding Embarked on top?



- Model [4]
  - Sex column (get_dummies(drop_first=True)), Pclass (raw), Embarked (categorized)
- Model [5]
  - Sex column (get_dummies(drop_first=True)), Pclass (raw), Embarked (get_dummies(drop_first=False))

Models #4 #5 - Analyzing the predictions

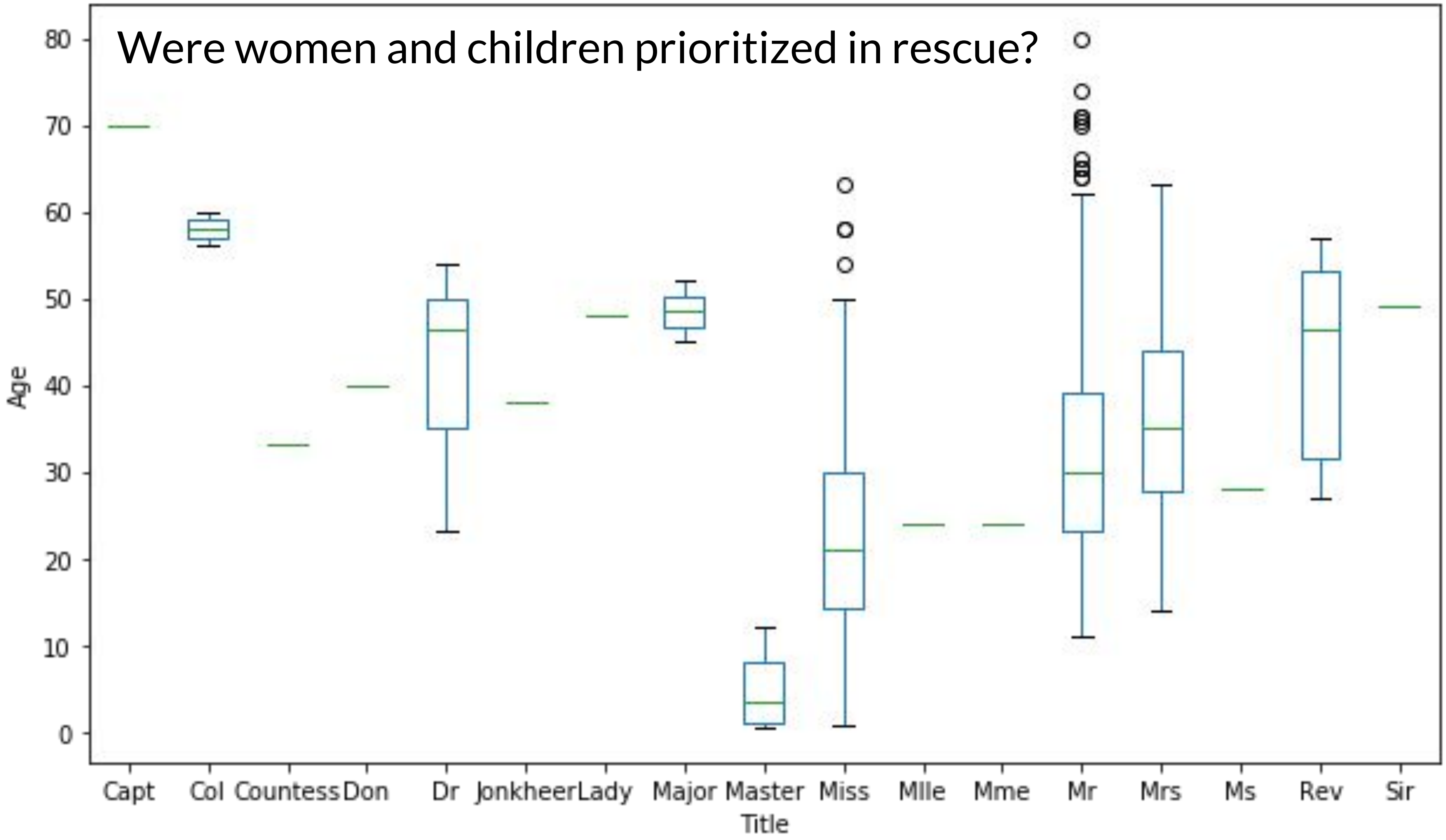


Model	Features	Parameters	Score	Ratio
All-Dead	-	All zeros	0.62679	-
[0] RandomForest	Sex (categorized) Pclass (raw)	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.74641	19.08%
[1] RandomForest	Sex (dummies(T)) Pclass (raw)	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.74641	19.08%
[2] RandomForest	Sex (dummies(T)) Pclass (dummies(F))	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.74641	19.08%
[3] XGBClassifier	Sex (dummies(T)) Pclass (raw) Age (dummies(F))	learning_rate: 0.001 max_depth: 4 n_estimators: 50	0.75119	19.84%
All-Females Survived All-Males Perished	-	if Sex == "female" Survived = 1 else Survived== 0	0.76555	22.13%
[4] RandomForest	Sex (dummies(T)) Pclass (raw) Embarked (Categorized)	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.77990	24.42%
[5] RandomForest	Sex (dummies(T)) Pclass (raw) Embarked (dummies(F))	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.77990	24.42%

The Title feature is a good predictor?

'Mr', 'Mrs', 'Miss', 'Master', 'Don', 'Rev', 'Dr', 'Mme', 'Ms', 'Major', 'Lady', 'Sir', 'Mlle', 'Col', 'Capt', 'Countess', 'Jonkheer', 'Dona'

Were women and children prioritized in rescue?



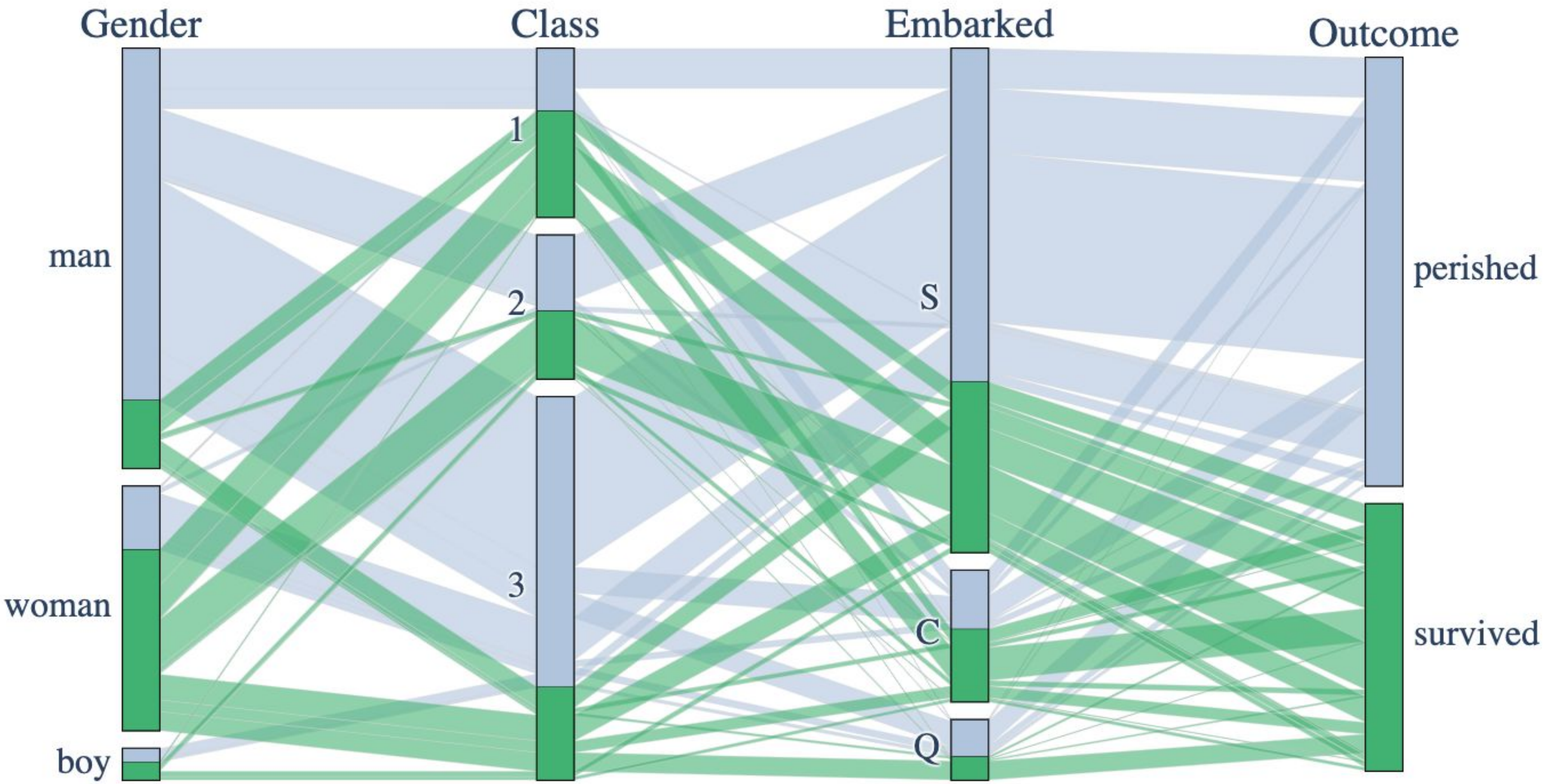
```python
df["Title"] = df["Name"].str.extract(' ([A-Za-z]+)\.',expand=False)
titles = {
    "Mr" :          "man",
    "Mme":          "woman",
    "Ms":           "woman",
    "Mrs" :         "woman",
    "Master" :      "boy",
    "Mlle":         "woman",
    "Miss" :        "woman",
    "Capt":         "man",
    "Col":          "man",
    "Major":        "man",
    "Dr":           "man",
    "Rev":          "man",
    "Jonkheer":     "man",
    "Don":          "man",
    "Sir" :         "man",
    "Countess":     "woman",
    "Dona":         "woman",
    "Lady" :        "woman"
}
```


New Sex Column

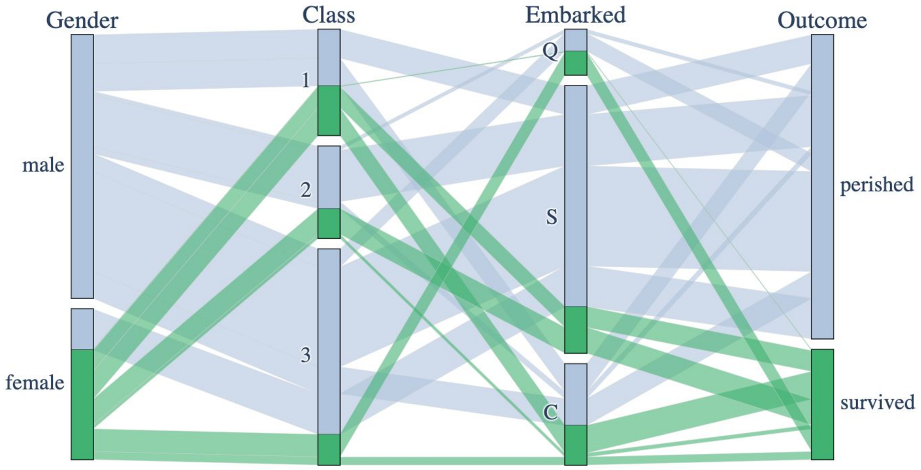
- sex+age+name

```python
df["Sex"] = df["Title"].map(titles)
```

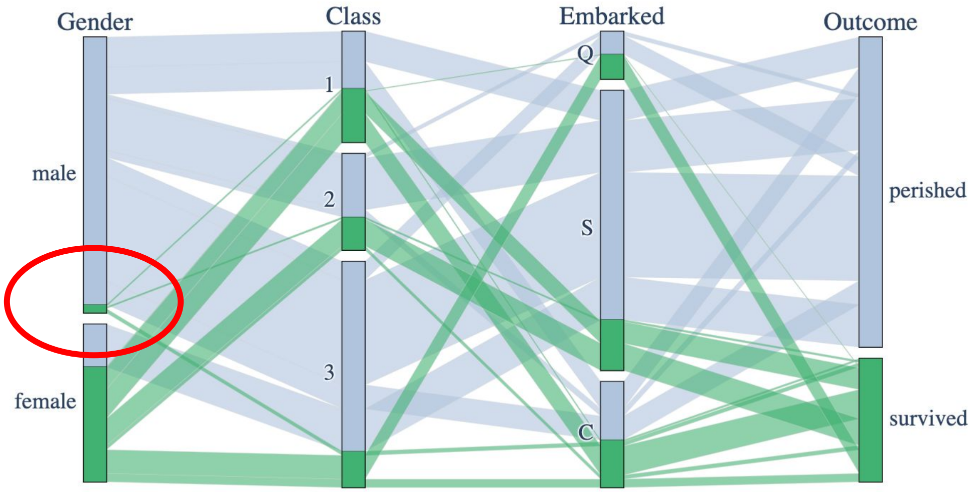
Models #6 #7 Gender = name+age



Models #4 and #5



Models #6 and #7



Model	Features	Parameters	Score	Ratio
All-Dead	-	All zeros	0.62679	-
All-Females Survived All-Males Perished	-	if Sex == "female" Survived = 1 else Survived== 0	0.76555	22.13%
[4] RandomForest	Sex (dummies(T)) Pclass (raw) Embarked (Categorized)	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.77990	24.42%
[5] RandomForest	Sex (dummies(T)) Pclass (raw) Embarked (dummies(F))	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.77990	24.42%
[6] RandomForest	Sex (name+age) Pclass (raw) Embarked (dummies(F))	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.78947	25.95%
[7] RandomForest	Sex dummies(name+age,T) Pclass (raw) Embarked (dummies(F))	criterion: entropy n_estimators:100 max_leaf_nodes: 64	0.78947	25.95%

Next Steps

- **Improving the features:**
  - Feature Engineering: Create new features from the existing data (family_size, ticket, cabin, fare, etc)
  - Feature Selection: Select the most relevant features to reduce noise and overfitting.
- **Improving the model:**
  - Model Selection: Try a variety of models to improve performance.
  - Hyperparameter Optimization: Optimize the settings within each particular machine learning model.

Getting Started with Kaggle.ipynb

https://www.kaggle.com/pliptor/how-am-i-doing-with-my-score/report