Report Titanic: Machine Learning from Disaster competition
by Capteam Iglo

Lisette Boeijenk (s1005856) - Tjalling Haije (s1011759) - Klaus Lux (s1012898)
Steven Smits (s4232763) - Zepp Uibo (s1012287) - Luuk Arts (s4396863)
Supervisor: Perry Groot

I. PROBLEM DESCRIPTION

The Titanic: Machine Learning from Disaster[1] is a getting started prediction competition, where the goal is to predict whether a passenger on board of the famous Titanic survived the disaster or not. Submissions are evaluated based on prediction accuracy (i.e. the percentage of passengers whose survival was correctly predicted.
There is a default split for the dataset, in which the training set contains 891 passengers. In the test set the survival label needs to be predicted for 418 other passengers. Information available in the datasets are for each passenger: the name of the passenger, ticket class, sex, age, number of siblings/spouses aboard, number of parents/children aboard, ticket number, passenger fare, cabin number and port of embarkation.

II. APPROACH

Python 3.5 was chosen as the language and the Machine Learning library Scikit-Learn[2] was used extensively.
The approach was to work iteratively on the essential elements of the classification task.

A. Literature Research

Literature research was done on how best to approach a Kaggle competition, since this was the first time for this team to do a Kaggle competition. Results showed for example that teams that worked iteratively and submitted early and often to Kaggle ended on higher positions[3], so the team strove for submitting early and often.
Kaggle kernels and the discussion board were consulted for additional ideas and literature research was done to find different models, to deal with the small amount of training data available, to find parameter optimization techniques and to answer more questions.

B. Pipeline

A pipeline was created to allow working iteratively on building out features and models without reshuffling the whole code base. Additionally, the pipeline was to make it easy to run the whole process using a simple command, from training the model to testing it and automatically generating the required predictions file to be uploaded to Kaggle.

[1] https://www.kaggle.com/c/titanic
[2] http://scikit-learn.org/stable/index.html
[3] https://rpubs.com/pedmiston/kaggle

C. Exploratory Data Analysis

Before starting any classification task, it is essential to do an exploratory data analysis (EDA) on the training data received. An EDA gives insights into what type of data is available, features that might be relevant, what preprocessing steps need to be taken and what type of models could be useful. EDA on the Titanic data showed:

- A strong correlation between sex and survival, with women being four times more likely to survive.
- A correlation between passenger class and survival, with first-class passengers being three times more likely to survive than third-class passengers.
- A similar correlation between the number of siblings and survival, higher numbers being associated with lower survival.

D. Preprocessing

Using insights from the EDA, preprocessing on the attributes of the Titanic that was found necessary. Steps taken in the preprocessing included replacing the Sex labels with a binary 0 or 1, filling in missing attribute data for Age and $Fare$ with the mean and $Embarked$ with the most frequent embarkation port. The resulting, complete $Embarked$ ports were replaced with numerical IDs and last the $PassengerId$ was removed, since instance IDs should not be used for classifications.

E. Feature Engineering

Literature showed that for most Kaggle competitions, feature engineering was more important and beneficial to high scores than picking which model to use. Feature engineering consisted of the subtasks feature generation and feature selection.

1) Feature Generation: For generating new features, Kaggle kernels and discussion boards were consulted and literature research was done, as well as any features were generated that the team could come up with. The generated features have been listed in Table I.

Moreover, a number of features were generated by binning features (e.g. age / fare) or combining them: For instance, the total number of relatives aboard was computed by adding the number of siblings / spouses to the number of parents / children.

Feature	Extraction
Fare per person	Identify groups by ticket number, divide fare by group size
Title (e.g. Mister, Miss)	Regex matching in Name
Ticket prefix	From Ticket feature
Deck number	From Cabin feature
Name length	From Name feature

TABLE I

FEATURES GENERATED AS PART OF THE FEATURE ENGINEERING.

2) Feature Selection: Feature selection is an important part of feature engineering, for training a model using features that are not significant can lower the accuracy drastically.
Features were selected using manual selection informed by different measures of feature informativeness, such as chi-squared and information gain: Features that were found not to be sufficiently predicted were not further considered. Depending on the model trained, the included features were varied (e.g. using the binned age feature only for Naive Bayes rather than both age features). For some models, features were additionally one-hot-encoded.

F. Model Implementation

As mentioned above, the Scikit-Learn library was used, which has diverse classification models available. The models implemented and explored for classifying Titanic survivors are: Adaboost, Bayes, Extra Trees, Genetic Programming, Gradient Boosting, K-Nearest Neighbor, Logistic Regression, Multi-Layer Perceptron, Random Forest, SVM and XGBoost.
For each of these models, different features were selected, Grid Search was applied to optimize the parameters[4] and 10-fold Cross Validation was used to estimate the model's accuracy for internal purposes[5]. Each individual model was used to predict survivors on the test set and uploaded to Kaggle. Additionally, each individual model was combined in the next step, into a ensemble model.

G. Ensemble Model Implementation

An ensemble model was generated from the individual models to compensate for over-fitting on the small training set as well as weaknesses of the individual models. A voting ensemble was implemented with options for soft voting (averaging) and hard voting (majority vote), as well as an option to add weighting based on model accuracy. The ensemble model was also subject to Grid Search to find the optimal parameters.
The resulting predictions from the ensemble model were also uploaded to Kaggle.

[4]Bayesian optimization was also considered for parameter optimization, but was found to be quite slow.

[5]The accuracy estimated by Cross Validation was generally lower than the Kaggle accuracy on the test set.

III. RESULTS

In total 125 submissions were uploaded to Kaggle. The highest leaderboard position reached was 210 out of around 11.000 teams[6], which places us in the top 2%. The accuracy for this submission of 0.82296 was reached with a majority vote voting ensemble of the Random Forest, K-nearest neighbor, Extra Trees, SVM, Logistic Regression and Naive Bayes models. Figure IV shows the cross validation values of each individual classifier model in the ensemble model, and shows that the KNN and Naive Bayes classifiers' accuracy is a bit lower, but the other accuracies lie close together. Figure IV shows the correlation between the predicted test set of each model, indicating the models differ in their prediction of instances. These differences make the ensemble model able to counterbalance some of the overfitting of the individual models (likely due to the small dataset, see section IV), and combine the correct predictions of the various models.

IV. DISCUSSION

There are a number of interesting observations that we made during this competition:

- Comparing different models, we found the accuracy differed little between them, however accuracy drops significantly when removing some generated features, indicating that feature selection and generation are more important than picking the right model for this competition.
- There does seem to be an inherent limit to the dataset when it comes to the maximum possible prediction accuracy. This is to be expected given that survival is very likely to be influenced by factors that are not or only very indirectly reflected by the socio-economic data, e.g. one's physical strength or health status. Other submissions substantially better than 85% are likely to involve some form of fraud due to the fact that the test set is publicly available.
- Due to the small dataset (≈900 training examples) there seems to be the issue of overfitting. This results in an unstable cross-validation accuracy estimation which overestimates the real accuracy on the Kaggle test set. The ensemble model does balance this overfitting slightly out.
- Submitting to Kaggle "early and often" as suggested by Pierce Edmiston in his analysis of successful Kaggle entries[7] proved to be an effective technique.

[6]Sunday the 15th of April.

[7]`https://rpubs.com/pedmiston/kaggle`

APPENDIX

Github repository: `https://github.com/thaije/capteam-iglo-titanic-ml`.

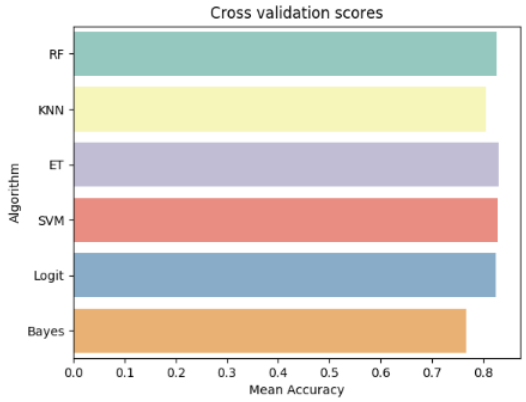

Fig. 1. Cross validation mean accuracy per model of best ensemble.

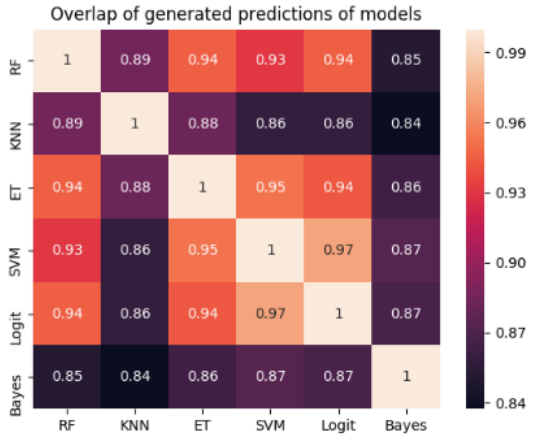

Fig. 2. Overlap between predicted test set of models of best ensemble.

TABLE II

CONTRIBUTIONS PER TEAM MEMBER

Team Member	**Contributions**
Klaus	Feature Engineering, Model Implementation, Parameter Optimization, Outlier Detection, Final Presentation
Lisette	Team Captain, Literature research, Model Implementation, Report, Final Presentation
Luuk	Model Implementation, Validation
Steven	EDA, Feature Engineering, Model Implementation, Ensemble Model Implementation
Tjalling	Preprocessing, Feature Engineering, Model Implementation, Ensemble Model Implementation, Flash Talk
Zepp	Pipeline, Model Implementation, Flash Talk