FIIPractic ML Project

Blanaru Dan

1. Introduction and tools

All the tools used in the making of this project are integrated in scikit-learn, pandas, seaborn or matplotlib. The only new package used is xgboost, but is often associated with the aforementioned. The dataset was “Titanic, Learning from Disaster”, freely available on kaggle as a competition dataset, but can be found fully from other sources.

I deleted much of the “code projection” phase while working because the separation is intended to be as modularized as possible, allowing each notebook to be a standalone presentation.

1. Preprocessing

I started by examining the data, trying to come up with new features and fixing the existing ones. Seeing as the only features I could think of are highly correlated to the existing ones, I preferred to only extract the title of the person from their name and conceptualize it into several classes.

The imputation state is simply attributing “S” to missing embarkings, as this is by far the most common value, and the mean to Fare, as I did not want to introduce any bias when associating a fare to a certain group. I also dropped Cabin, as it contained way to many null values and most algorithms don’t do well with NaN’s and the information gain was small due to it, other features providing the same information that would require hard work to be extracted from this column.

Two encoding options can be used, one hot or ordinal. Altough a one-hot encoder must be passed as a parameter, I did not use it seeing that

Standardization simply demeans and divides by standard deviation for the columns which are continuous.

1. Visualisation

I sticked to plots that show actual information, not just look pretty, this still gave me a wide array of possibilities, and the ones I included in the notebook I think are the most relevant. I think the graphs show the general trends in the dataset and all the correlations discovered make sense.

1. PCA as a visualization tool

In this context(having 20-or so dimensional data) PCA doesn’t show its strengths as a dimensionality reduction for memory or computational efficiency. However, I used it to visualize the transformations I used and figure out how complex was the dataset given a particular encoding.

1. Modelling

Although this may take the longest to fully develop, the tools available from scikit-learn actually made it very easy. The code in this part is perhaps the most complex and least commentated, but all I did was create some wrappers around the scikit-learn API and train all the models all at once to see which of them had a good baseline to improve upon. While I tried any grids for some of them, none really helped me get past 80%.

Thank you very much for your time and sharing the knowledge, hope to see you soon!