Data Mining - Assignment 1., report

Wojciech Nagórka, Andrii Chmutov, Vasyl Korzavatykh, Kuba Czech

May 9, 2024

1 Introduction

This report sums up a work about usage of data pre-processing techniques to improve accuracy of algorithms working on the dataset. Report contains following sections:

- Description of dataset
- Description of input features
- Analysis of the input features
- Description of pre-processing techniques used in the assignment with:
  - Description
  - Motivation
- Description of output features
- Exploratory description of output features
- Conclusions

Authors of the project: Wojciech Nagórka, Andrii Chmutov, Vasyl Korzavatykh, Kuba Czech

2 Description of Dataset

Dataset chosen by our group is Titanic, which is well-known dataset used very often in machine learning and data analysis. This dataset contains information about 891 passengers aboard the RMS Titanic, which sank on its maiden voyage after hitting an iceberg in April 1912.

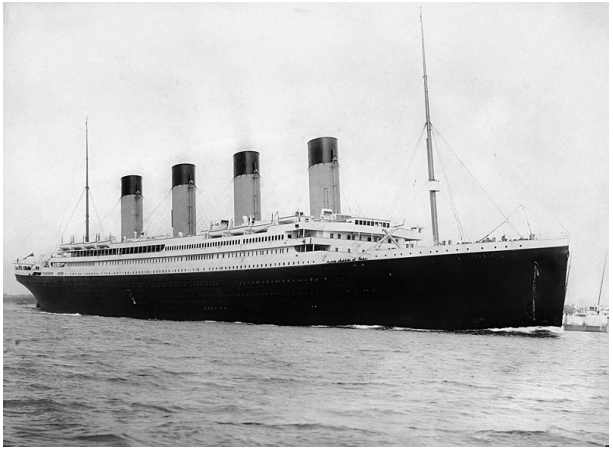


Figure 1: RMS Titanic before sinking

Titanic dataset is composed of following features: PassengerID, Survived, Pclass, Lname, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked which will be described in more detailed way later in this report.

3 Description of input features

3.1 PassengerID

This feature represents an ID of each passenger, which is unique, integer number starting with 1.

3.2 Survived

This feature contains a boolean variable representing status of the passenger - whether he/she survived or died.
The most important fact is that this is is target variable.

3.3 Pclass

This feature contains an integer number from 1 to 3 representing socio-economic status of the passenger:

- 1 stands for upper class
- 2 stands for middle class
- 3 stands for lower class

3.4 Lname

This feature contains a string representing passenger's last name

3.5 Name

This feature contains a string representing passenger's first name with one's title

3.6 Sex

This feature contains a string ('male' or 'female') representing passenger's sex.

3.7 Age

This feature contains a float number representing passenger's age.

3.8 SibSp

This feature contains integer number representing number of siblings or spouses aboard Titanic for each passenger

3.9 Parch

This feature contains integer number representing number of parents or children aboard the Titanic for each passenger

3.10 Ticket

This feature contains string representing passenger's number of ticket.

3.11 Fare

This feature contains float number representing how much each passenger paid to buy his/her ticket.

3.12 Cabin

This feature contains string representing number of cabin of each passenger. First character of cabin's number represents deck where cabin was located, rest of this string is room's number.

3.13 Embarked

This feature contain string representing port of embarkment of each passenger:

- C stands for Cherbourg
- Q stands for Queensland
- S stands for Southampton

4 Exploratory description of input features

4.1 Checking types of features

First let's have a look at types of data in each feature (whether they are boolean, int or string, how many features (columns) and how many row entries do we have.

```
<class 'pandas.core.frame.DataFrame'>
Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       891 non-null    object
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64
 6   Parch     891 non-null    int64
 7   Ticket    891 non-null    object
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object
 10  Embarked  889 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 83.5+ KB
```

4.2 Basic statistical data describing numeric attributes

Now let's describe dataset using mean, standard deviation, min, first quartile, second quartile, third quartile and max

	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

4.3 Missing values

Potential problem we may also encoutner is Nan values - they may exist in *Age*, *Cabin* and *Embarked* attribute. Let's have a look at how many cases of missing data do we have

```
Survived      0
Pclass        0
Name          0
Sex           0
Age         177
SibSp         0
Parch         0
Ticket        0
Fare          0
Cabin       687
Embarked      2
dtype: int64
```

4.4 Unique values

We can also check uniqueness of the values - for sure it will be a good practice to know your data!

	Unique values	Data type
Survived	2	int64
Pclass	3	int64
Name	891	object
Sex	2	object
Age	88	float64
SibSp	7	int64
Parch	7	int64
Ticket	681	object
Fare	248	float64
Cabin	147	object
Embarked	3	object

We see that for example in attribute *Embarked* number of unique values is small and equals 3 (we don't take into account missing values). We know that it is because there were three ports of embarkment. Very similar applies to *Sex* or *Pclass* because those are discrete values. Other interesting fact is that number of unique cabins is equal to 147, which is only partially true - after dropping Nan values we were left with only 207 valid records.

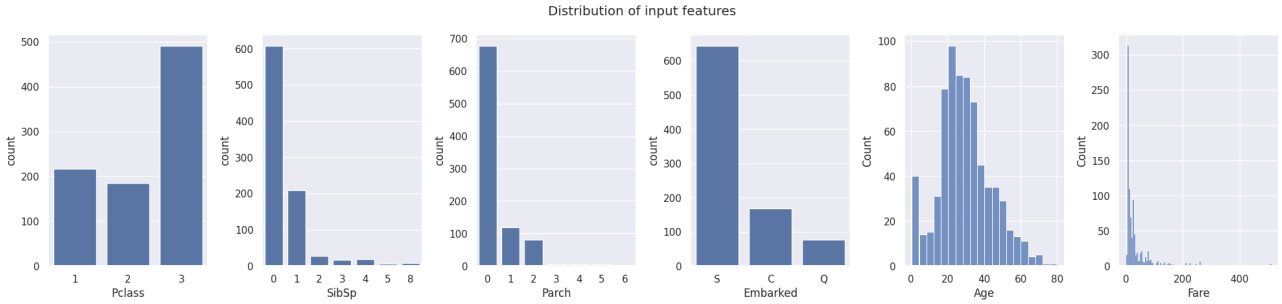
4.5 Combination of comparisons between attributes

Next thing we can do is to compare every two attributes with respect to status of the passenger (died or survived).

Of course we don't compare every pair of attributes because it is useless to compare name (all values are unique) with port of embarkment.

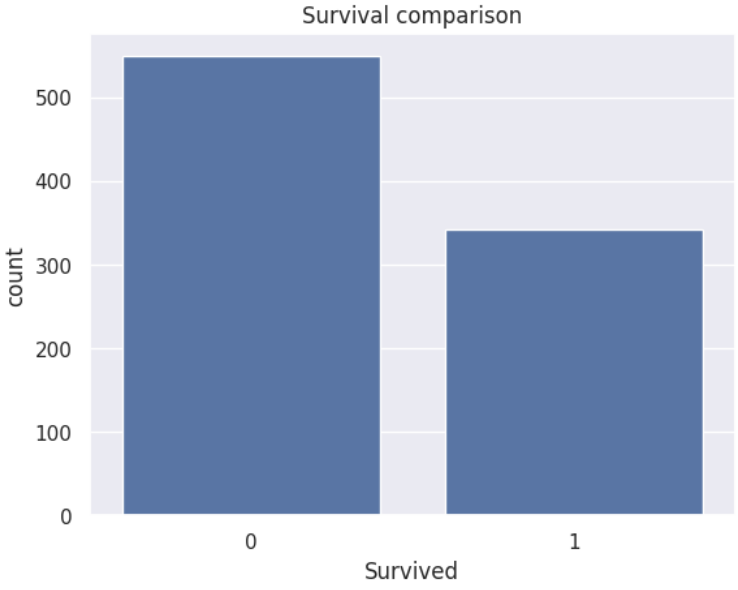
4.6 Distribution of input features

Now we can look more precisely at distribution of input features.

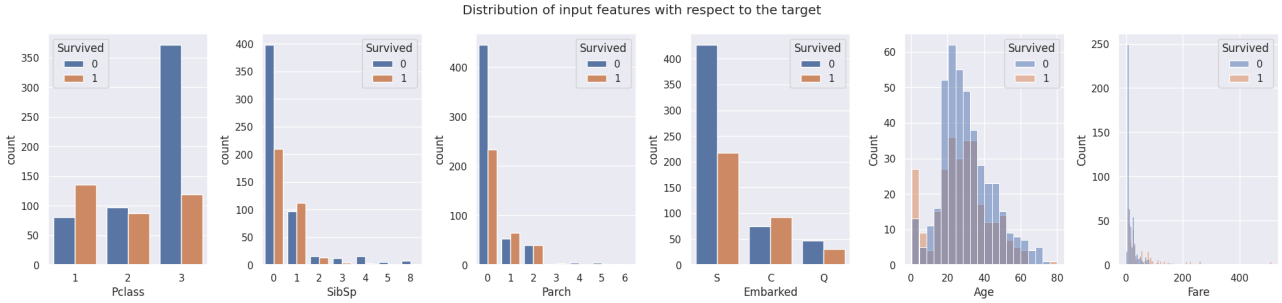


4.7 Distribution of target variables

Let's also see how many of the passengers died or survived



And how does it depend on other attributes



5 Description of pre-processing techniques used in the assignment

5.1 Standardization

Titanic dataset offers different types of data (strings, boolean and continuous data having wide range of values). Potentially, continuous data in its raw form may cause some problems and require to be converted into something else and one of the methods to overcome this problem is to standardize them. We used StandardScaler() method from sklearn library, which easily transforms input data into data with mean equal to 0 and standard deviation equal to 1 and applied it to *Age* and *Fare* attribute. After this operation, the *Age* is in more reasonable range instead of ranging from 0.42 to 80. Very similarly, *Fare* doesn't range from 0 to 512.3 anymore - we can clearly see that advantage of this technique is that it handles the outliers.

5.2 Imputation of missing values

Another problem we had to take into account was missing values (Nan). Although it happened only in three attributes (*Age*, *Cabin* and *Embarkment*) it produces serious problem because Nan values algebra could produce strange results. Our approach was to substitute every missing value with mean, median or mode (most frequent), depending on parameters provided by the user. After that operation, dataset doesn't contain Nan values and future pre-processing can be applied.

5.3 Derived attributes

As previously mentioned, it sometimes happens that important information is hidden in the data (deck or title). We decided to create our own attributes by processing proper information from specific attributes. It resulted in creating new attributes such as *Deck* or *FamilySize* and as an input to create new features (*Alone* or later label coding).

5.4 Binarization

Sometimes more important than exact value of the attribute, is whether it fulfills the condition or not. This is the case with attribute *Alone* - more important than exact size of the family is whether it is equal to zero or not. As a result we get boolean value.

5.5 Label encoding

It may happen that we would need to convert value of the attribute into entities. This is the case with attributes *Sex*, *Embarkment* and title extracted from name. It is achieved by pandas function get_dummies() and as result we get one-hot encoding.

6 Description of output features

We already covered description and analysis of input features, now it is time to say few words about output features.

6.1 Features without change

It is not necessary to pre-process every feature and it is not the best option to change target variable in any way. Attributes that remained unchanged are: *PassengerID*, *Survived* (target variable) and *Pclass*.

6.2 Age

Values of this attribute are standardized values of input feature *Age*. As a result we get values with mean 0 and standard deviation equal 1.

6.3 Fare

Similarly as with Age, values of this feature are standardized values of input feature *Fare*.

6.4 Deck

As mentioned above, deck can be extracted from cabin number (first character) and later it is converted into ordinal numbers.

6.5 FamilySize

This feature contains information about size of family of each passenger stated as integer number.

6.6 Alone

This feature contains information whether a passenger travelled alone or not (i. e. if FamilySize is equal to zero) stated as a boolean variable.

6.7 Embarked_C, Embarked_Q, Embarked_S

In paragraph describing techniques of data pre-processing used in this project, there was information about label encoding. Those three columns (boolean variables) represent port of embarkment of a passenger and this three attributes substitute previous attribute *Embarkment*.

6.8 Title_Master, Title_Miss, Title_Mr, Title_Mrs, Title_Rare

Similarly to three previous features, those attributes stated as boolean variables represent title of each passenger (Master, Miss, Mr, Mrs) or if a title appears rarely (for example 2 times per 891 passengers) it is assigned as Rare.

6.9 Sex_male, Sex_female

The same as with titles and with port of embarkment, those two features represent sex of each passenger stated as boolean variables.

7 Exploratory analysis of output features

7.1 Basic stats of numerical features of preprocessed data

First of all, let's take a look at basic stats (min, max, mean etc) of preprocessed data.

	Survived	Pclass	Age	Fare	Deck	FamilySize
count	891.000000	891.000000	8.910000e+02	8.910000e+02	891.000000	891.000000
mean	0.383838	2.308642	2.272780e-16	3.987333e-18	6.716049	0.904602
std	0.486592	0.836071	1.000562e+00	1.000562e+00	2.460739	1.613459
min	0.000000	1.000000	-2.224156e+00	-6.484217e-01	0.000000	0.000000
25%	0.000000	2.000000	-5.657365e-01	-4.891482e-01	8.000000	0.000000
50%	0.000000	3.000000	-1.046374e-01	-3.573909e-01	8.000000	0.000000
75%	1.000000	3.000000	4.333115e-01	-2.424635e-02	8.000000	1.000000
max	1.000000	3.000000	3.891554e+00	9.667167e+00	8.000000	10.000000

As previously mentioned, mean of standardized parameters (*Age* and *Fare*) equals 0 and standard deviation equals 1. However, more important fact is that our standardized features are not that much affected by outliers.

7.2 Number of Nan values

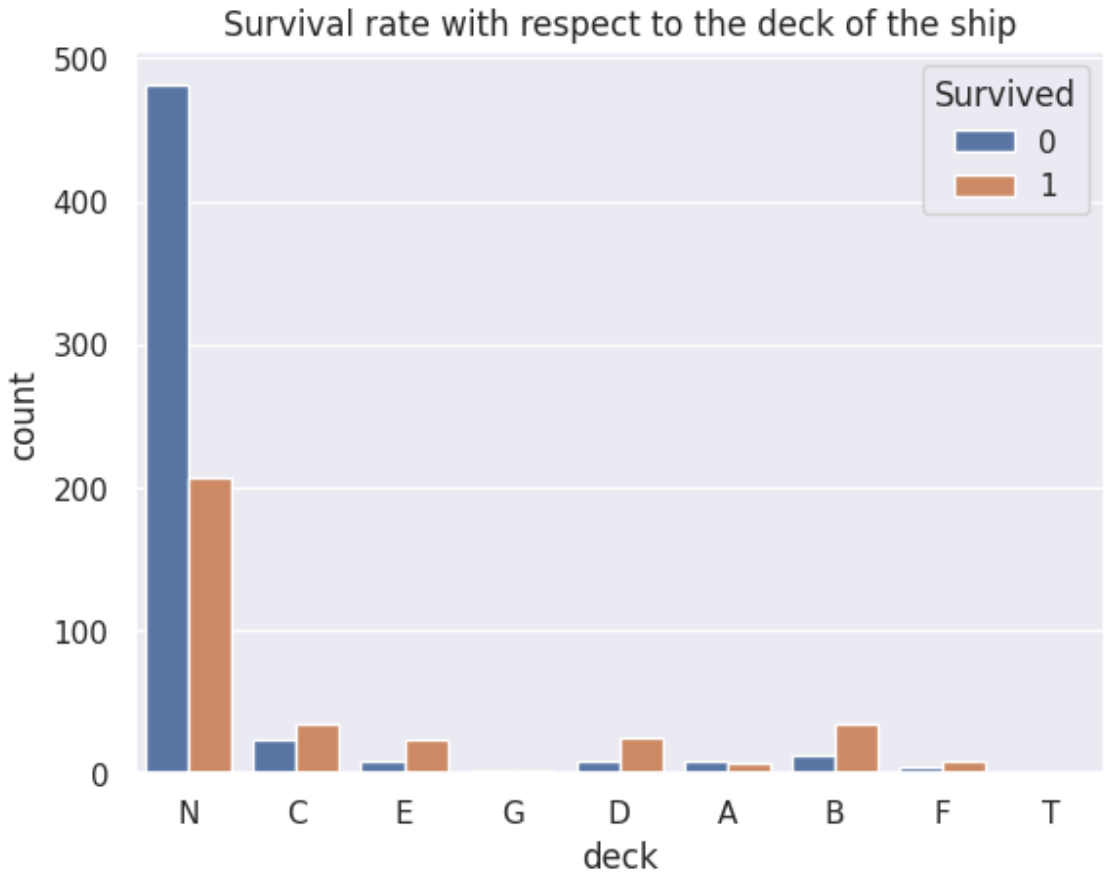
Another important thing to mention is lack of Nan values in our preprocessed dataset. Missing values were substituted in 3 atributes and thus they are not a problem for us anymore.

Screenshot to be inserted

7.3 Decks vs passenger status

Of course above plots and analysis is not enough because we have some information "hidden" i. e. some parameters were derived from input features. Example of such is deck where passenger's cabin was located - let's have a closer look at what are the decks and how does survival rate depend on it.

```
array(['N', 'C', 'E', 'G', 'D', 'A', 'B', 'F', 'T'], dtype=object)
```



Not to much information was provided by this plot - we see high bars for 'N' category (which is letter used to express that cabin and thus deck are not know). Other interesting fact is that the relatively high survived to died rate was on 'B', 'D' and 'E' deck.

7.4 Dependancy between title and survival status

Another hidden feature was occupied title (Mr., Mrs. etc) - we can analyze how status of passenger depend on this information.

Name	
Mr	517
Miss	182
Mrs	125
Master	40
Dr	7
Rev	6
Col	2
Mlle	2
Major	2
Ms	1
Mme	1
Don	1
Lady	1
Sir	1
Capt	1
the Countess	1
Jonkheer	1

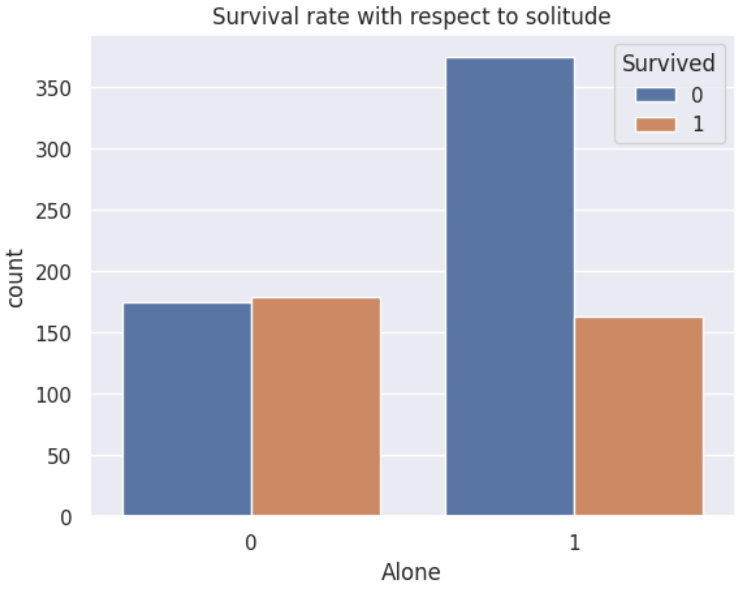
And we can see some interesting pattern - when comparing the titles, we see that for 'Mr' title there is very high blue bar compared to orange bar (in other words a lot of men died when Titanic sank and only few survived) compared to women (Miss title, Master title standind for young boy and most importantly Mrs title standing for married women) where orange bars are higher than blue ones (more were rescued). We can say that this information is no big surprise, because in most cases of emergency, women have priority over men.

7.5 *FamilySize* and *Lonely* attributes

Last factor we will take into account is family and we will accomplish it by utilising *FamilySize* and *Lonely* attributes. Let's first start with size of family (just to remind - sum of *Sibsp* and *Parch* attributes which is number of spouses/partners aboard and parents/children).

We see that members of family had higher chances of survival but if family is relatively small (up to 3 members), which can be treated as a bit counter-intuitive.

Interesting statistics is whether passenger was travelling lonely or not (now we are not interested how big was size of the family but whether the passenger was with someone or not).



Once again, the outcome isn't a big surprise and was partially visible on previous plot - bigger chance of survival had people that weren't travelling alone (however the chance of surviving when travelling with someone was 50% as bars are almost equal). The reasoning behind it, is that very often parents (especially mothers) with children were evacuated before everyone else and passengers not accompanied by anybody had smaller priority.

7.6 Improved Accuracy

The most important thing however, is accuracy of prediction for pre-processed data. Overally it improved but size of improvement depends on the test set. It is presented in tables below:

```
SVC
Scores: [0.68, 0.69, 0.67, 0.67, 0.7]
Mean score: 0.68

Perceptron
Scores: [0.63, 0.53, 0.61, 0.71, 0.6]
Mean score: 0.62

Random Forest
Scores: [0.84, 0.81, 0.76, 0.84, 0.83]
Mean score: 0.82
```

Figure 2: Accuracy of prediction for data without pre-processing

```
SVC
Scores: [0.83, 0.8, 0.8, 0.84, 0.84]
Mean score: 0.82

Perceptron
Scores: [0.82, 0.8, 0.76, 0.69, 0.87]
Mean score: 0.79

Random Forest
Scores: [0.84, 0.8, 0.8, 0.85, 0.85]
Mean score: 0.83
```

Figure 3: Accuracy of prediction for data with pre-processing

As previously mentioned, improvement is not the same - for SVC it was 15 percentage points but for Random Forest only 1 percentage point (yet, still improved).

8 Final conclusions

To sum up, used pre-processing techniques (standardization, binarization, label encoding etc) improved accuracy of classification on the dataset - this only highlights its importance in data analysis.