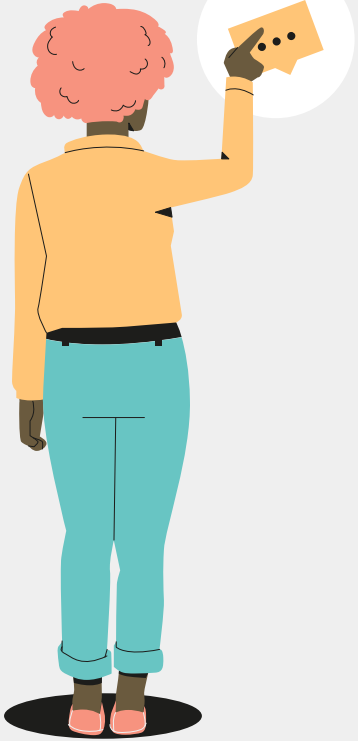


Data Mining
Assignment 1

Data pre-processing

done by - Lidia Wiśniewska - Joanna Szczublińska -
- Maria Musiał - Wiktoria Szarzyńska -

Main idea of the assignment and our goal.

- Main idea: to process dataset using using a set of pre-processing algorithms from the fields of feature normalization and standardization, feature selection, feature extraction.
- Our goal: to improve the general classification result and to predict if given passenger will survive the Titanic catastrophe.
- Our dataset: Titanic Dataset.

Strategy - Assignment Step by Step

Pre-processing

Analyzing

Choosing dataset - classification and regression problem

Normalization

Process of organizing data in a database

Selection

Selecting the best model for a particular problem

Extraction

Process of retrieving data out of data sources for further data processing or data storage

00 - Let’s talk about chosen Dataset.

- Titanic Dataset: describes the survival status of individual passengers on the Titanic.
- Titanic attributes:
  passengerID - unique passenger number, Pclass - class (first, second, third), Name, Sex, Ticket - number of ticket, Fare - fee of the [illegible] Cabin, Embarked - starting port, Parch - number of parents Sibsp - number of siblings or spouse.

00 - Let’s talk about chosen Dataset.

PassengerId	Survived	Pclass	Name	Sex	Age
153	0.0	3	Meo, Mr. Alfonzo	male	55.5
472	0.0	3	Cacic, Mr. Luka	male	38.0
1233	NaN	3	Lundstrom, Mr. Thure Edvin	male	32.0
60	0.0	3	Goodwin, Master. William Frederick	male	11.0

PassengerId	SibSp	Parch	Ticket	Fare	Cabin	Embarked
153	0	0	A.5. 11206	8.0500	NaN	S
472	0	0	315089	8.6625	NaN	S
1233	0	0	350403	7.5792	NaN	S
60	5	2	CA 2144	46.9000	NaN	S

01 - Before PreProcessing

Before normalization: analyzing our data

Checking accuracy, searching and brainstorming

Accuracy at the start, it's values

Names of columns, null values, info about rows

Using functions such as: normalizeData and ChangeNullValues

Using new class to organize our data

First step and what changed

01 - Before PreProcessing

```
def changeNameToSurname(data, column):


    for index, row in data.iterrows():
        surname = row.Name.split(',')[0]
        data.at[index, column] = surname
```

Additionally:
changeCategoricalToDiscrete and
changeContinousToDiscrete

```
def changeNullValuesToMean(data):
    columns_names = data.columns.to_list()
    for column in columns_names:
        value = None
        null_percentage = data[column].isnull().mean()

        if null_percentage > 0.05:

            if data[column].dtype == 'object': ##for caterogical type
                value = data[column].mode()[0] ### takes first one
            else:
                value = round(data[column].mean(),2)

        if value != None:
            for index, row in data.iterrows():
                if pd.isnull(row[column]):
                    data.at[index, column] = value
```

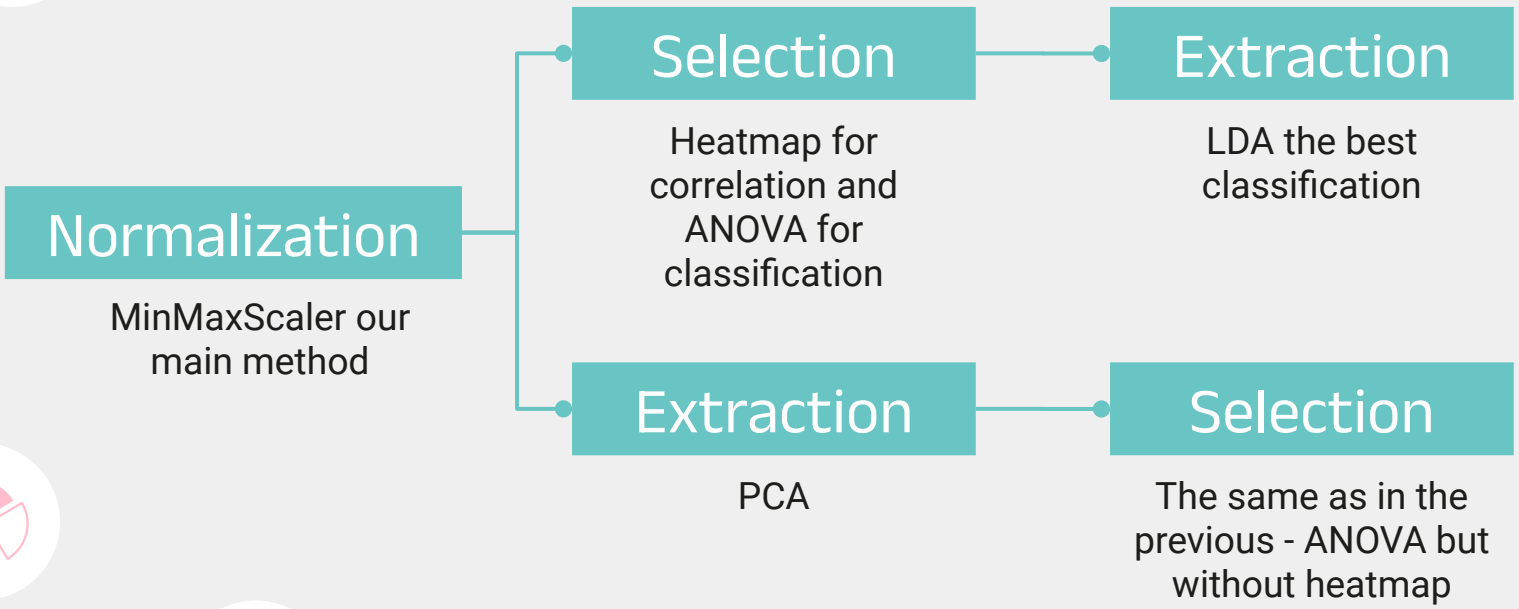
01 - Let’s talk about analyzing our data

- Changes the 'Name' column in the DataFrame to contain only surnames.
- Discretized continuous values in columns using KBinsDiscretizer.
- Encodes categorical columns from dataFrame using LabelEncoder.
- Replaces null values with mean or mode for columns with null percentage > 0.05.

01 - Let’s talk about analyzing our data

- First method: split to train and data set, null values as mean, mode, change categorical to discrete
- Second method: split to train and data set, null values as mean, mode change categorical to discrete, assign Fare and Age to 3 classes
- Third method: split to train and data set, null values as mean, mode change categorical to discrete, assign Fare and Age to 5 classes (the same accuracy for 8 classes)

02 - Preprocessing algorithms

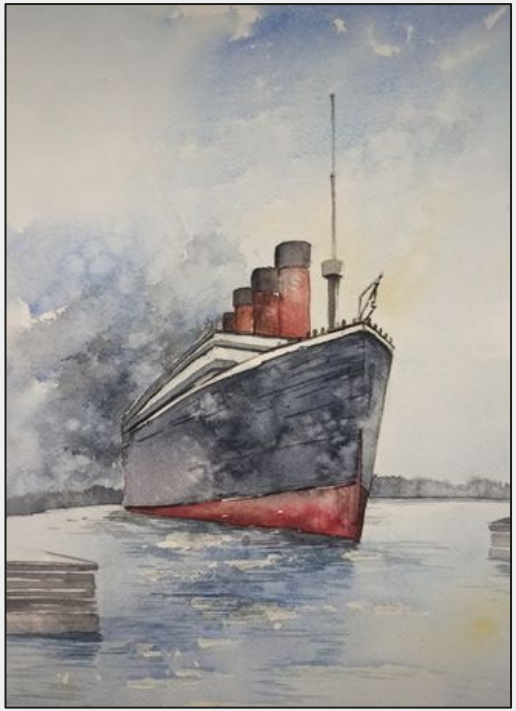


2.1 - Normalization

```
def normalizeData(data , target_column):

target_column_index = data.columns.get_loc(target_column)

    X = pd.concat([data.iloc[:, :target_column_index], data.iloc[:, target_column_index+1:]], axis=1)

    cols_all = data.columns.tolist()
    cols = cols_all[:target_column_index] + cols_all[target_column_index+1:]

    norm = MinMaxScaler(feature_range=(0,1)).fit(X)
    normalized_data = pd.DataFrame(norm.transform(X), columns=cols)

    return normalized_data
```

2.1 - Let’s talk about Normalization

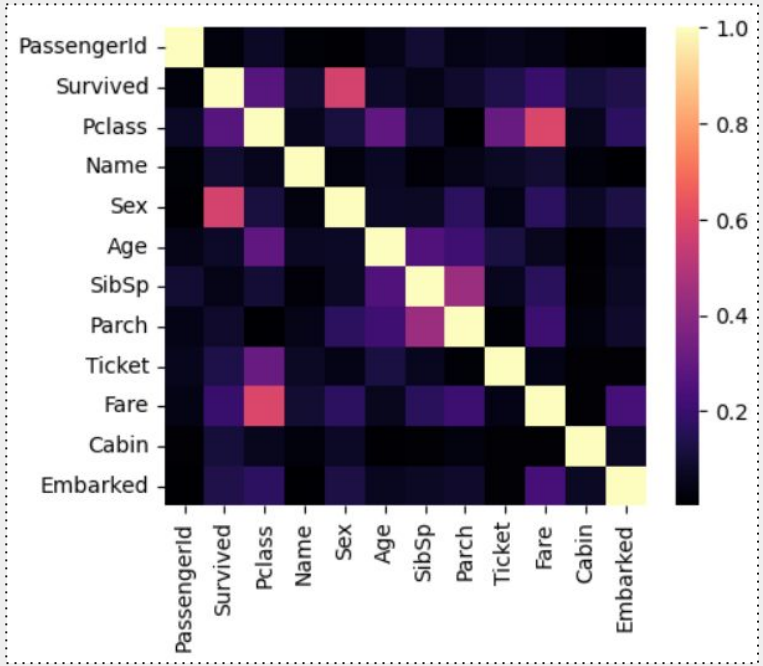
- Normalizes the data using Min-Max scaling from sklearn
- First attempt accuracy - 0.78991

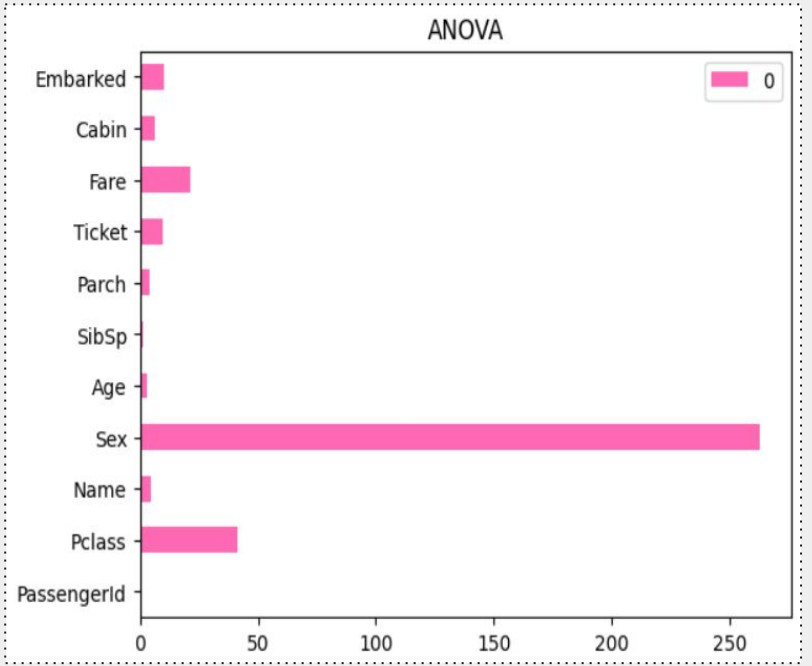


2.1 - Feature Selection

```
def computeAnova(normalized_data, target_values):

    cols = normalized_data.columns.tolist()
    scores_anova, p_vals_anova = f_classif(normalized_data, target_values)
    dataframe = pd.DataFrame(scores_anova, cols)

    return dataframe
```

2.1 - Feature Selection





2.1 - Let’s talk about Feature Selection

- Computes ANOVA scores for feature selection based on normalized data
- First attempt accuracy - 0.8039

2.1 - Let’s talk about Feature Extraction

- LDA - projects high-dimensional data to a low-dimensional space with discriminative features
- First attempt accuracy - 0.787

2.2 - Normalization

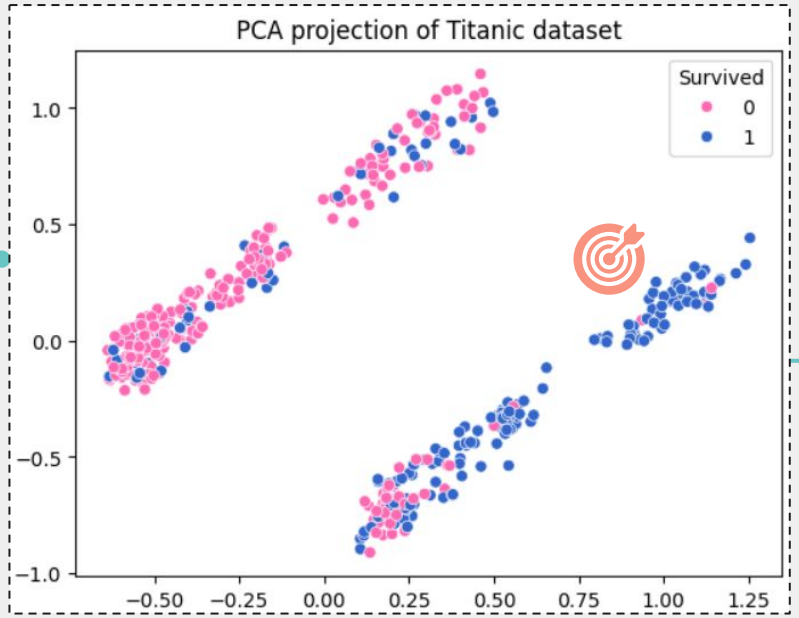
```
clas.changeNullValuesToNewValue(test_df)
clas.changeNameToSurname(test_df, 'Name')
clas.changeCaterogicalToDescrite(test_df)
clas.changeContinuousToDescrite(test_df)
normalized_data_test = clas.normalizeData(test_df, 'Survived')

X = normalized_data
y = train_df['Survived']

X_test = normalized_data_test
y_test = test_df['Survived']

acc = clas.getAccuracy(X, y, X_test, y_test)
print("Accuracy after 1 step of preprocessing: ", acc)
attempt_2.append(acc)
```

2.2 - Feature Extraction

```
def computePCA(X, X_test):

    pca_transformer = PCA(n_components=4)

    X_pca = pca_transformer.fit_transform(X)
    X_pca_test = pca_transformer.transform(X_test)
    pca_df = pd.DataFrame(X_pca)
    pca_df_test = pd.DataFrame(X_pca_test)
    pca_components = pca_transformer.components_
    print("PCA components: ", pca_components)

    return X_pca, X_pca_test, pca_df, pca_df_test
```

2.2 - Feature Extraction

Dimensionality reduction

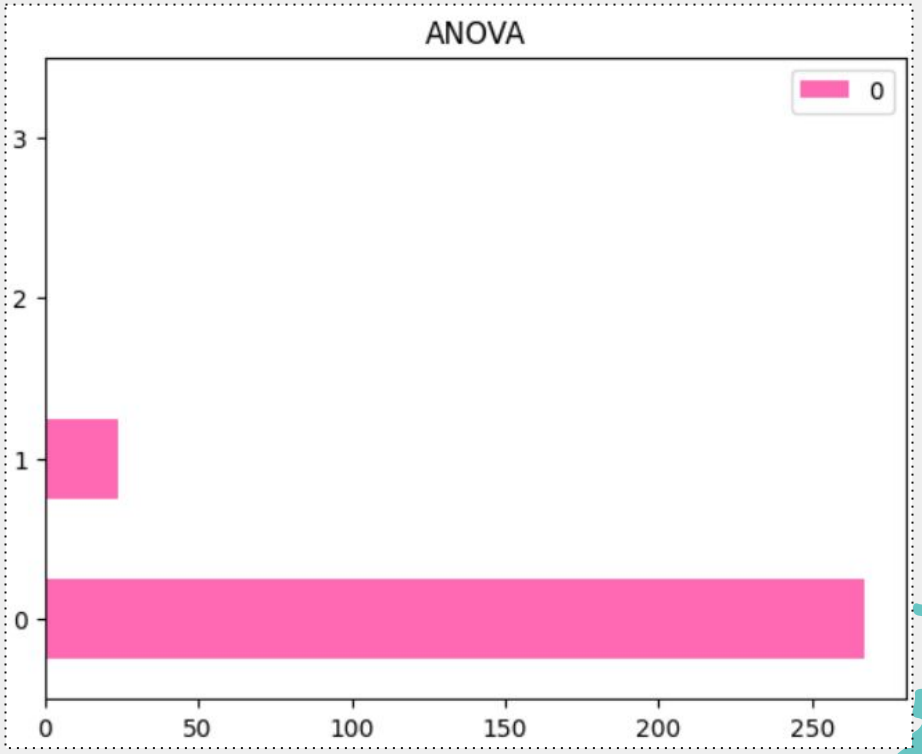


Noise reduction

2.2- Let’s talk about Feature Extraction

- Dimensionality reduction - which let transforms the original dataset into a smaller set of attributes (features) where retain most of the variance in the data. It helps to avoid overfitting in high dimensional dataset such as us and let us make faster computation of this set in future
- Noise reduction - which help to reduce the impact of noisy(irrelevant) features on the classification task. Therefore selected feature space contain the most information

2.2 - Feature Selection



2.2- Let’s talk about Feature Selection

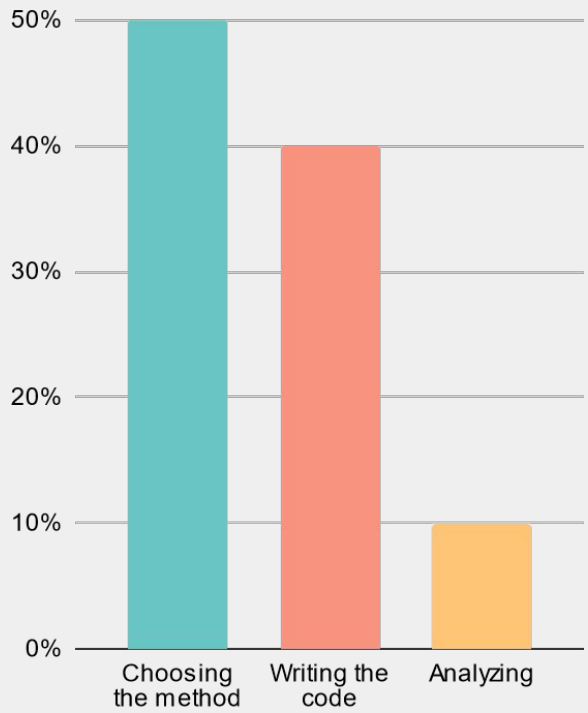
- Accuracy after preprocessing: 0.82072
- Summary: then we do ANOVA test as earlier from all 4 componentes the most important are '0', therefore we leave them in our dataset and drop others, our accuracy increased
- if from all we choose '0' and '1'

  (we get 0.76750)

04 - To Sum Up

	Attempt 1	Attempt 2	Summary
Before	**69.7%**	**69.7%**	Initial accuracy
1 step	**78.9%**	**78.4%**	Difference is small
2 step	**80%**	**78%**	Difference is quite high
3 step	**78.7%**	**82%**	The best - attempt number 2

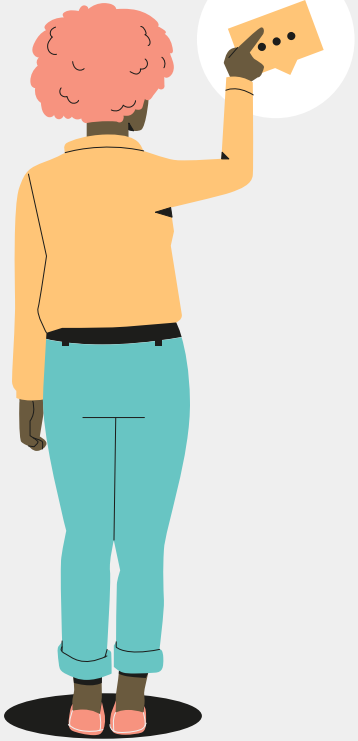
Summary - time and commitment



50% Choosing the method

40% Writing the code

10% Analyzing the input



Thank You

For your attention

Data pre-processing

done by - Lidia Wiśniewska - Joanna Szczublińska -
- Maria Musiał - Wiktoria Szarzyńska -