A Project Report On

LP3 – Machine Learning

CLASS: BE-1

GUIDED BY

Prof. Shweta Shah

DEPARTMENT OF COMPUTER ENGINEERING

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

DHANKAWADI, PUNE-43

SAVITRIBAI PHULE PUNE UNIVERSITY

2021-22

Gayatri Godbole – 41128

Pushkar Jain – 41134

Title:

Apply machine learning on Titanic Dataset

Problem Definition:

Build a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data (i.e. name, age, gender, socio-economic class, etc.).
Dataset Link: https://www.kaggle.com/competitions/titanic/dataObjective

Hardware Requirements:

4gb Ram

64 bit OS

Software Requirements:

Jupyter Notebook

Python

Windows/Ubuntu

Theory:

About Titanic Dataset-

The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered “unsinkable” RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren’t enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

In this problem, we build a predictive model that answers the question: “what sorts of people were more likely to survive?” using passenger data (ie name, age, gender, socio-economic class, etc).

Data Overview

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

KNN Algorithm

In statistics, the *k*-nearest neighbors algorithm (*k*-NN) is a non-parametric supervised learning method first developed by Evelyn Fix and Joseph Hodges in 1951, and later expanded by Thomas Cover. It is used for classification and regression. In both cases, the input consists of the *k* closest training examples in a data set. The output depends on whether *k*-NN is used for classification or regression:

- In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.

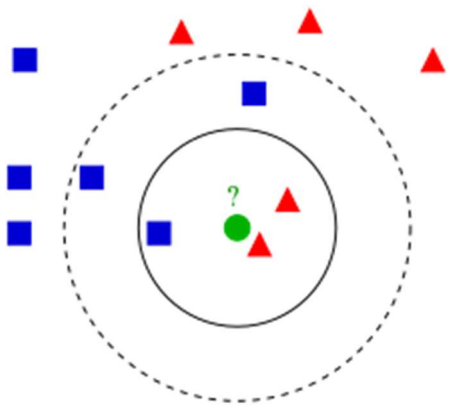
- In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

k-NN is a type of classification where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically.

Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/d$, where d is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

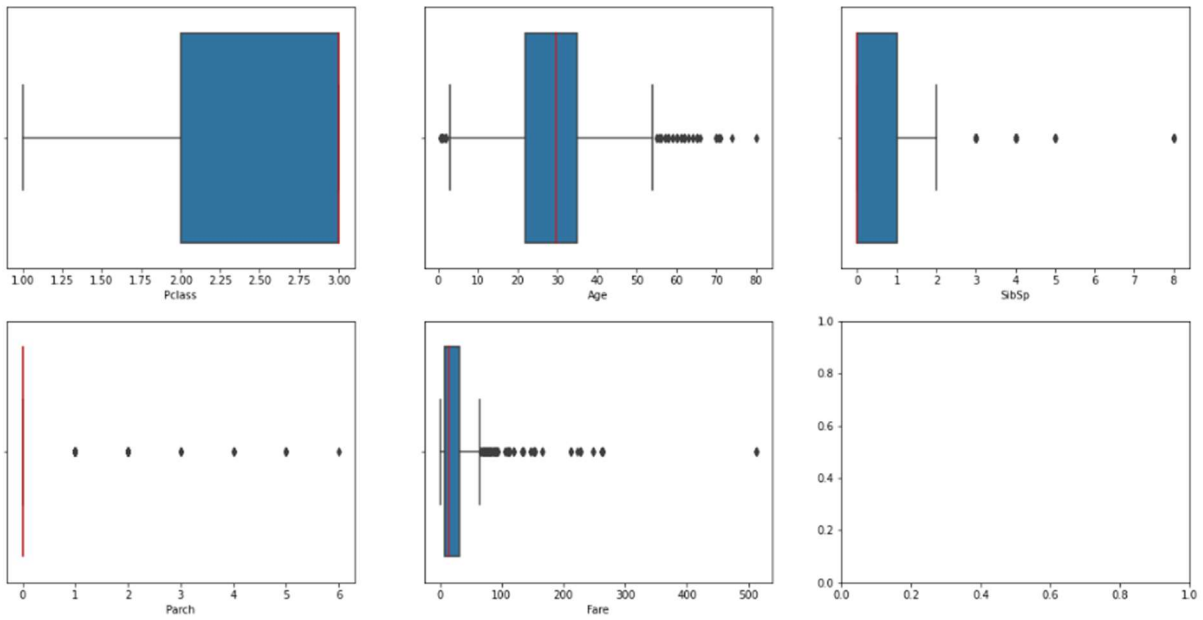


Methodology

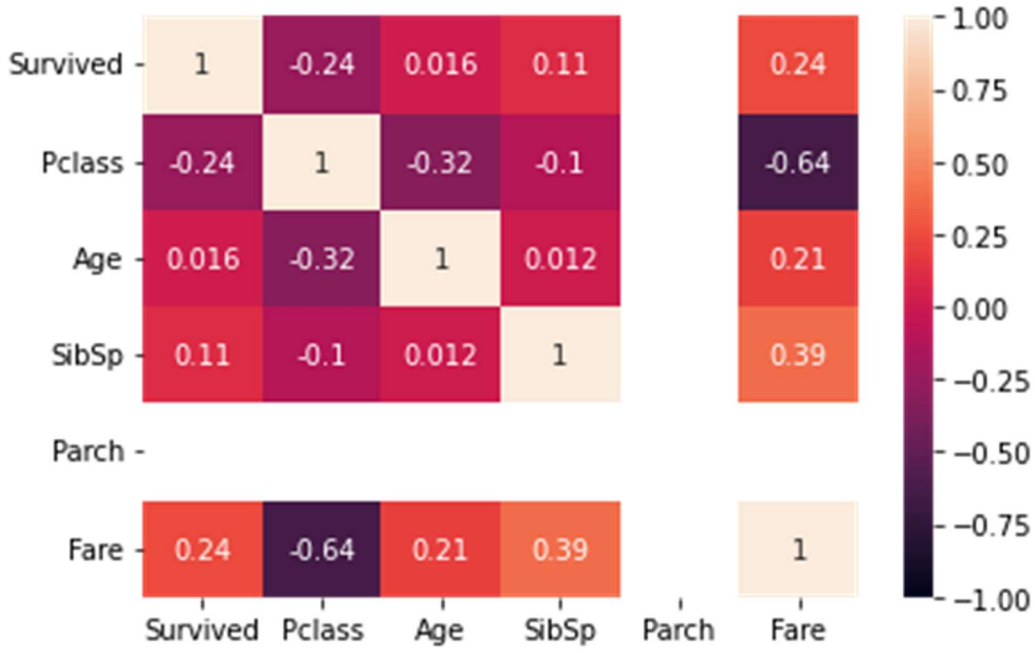
1. Load essential libraries
2. Read the dataset
3. Perform EDA
4. Check outliers
5. Remove if any
6. Choose a model
7. Normalise data of needed for model
8. Train model
9. Test model
10. Plot accuracy metrics

Output Screen Shots

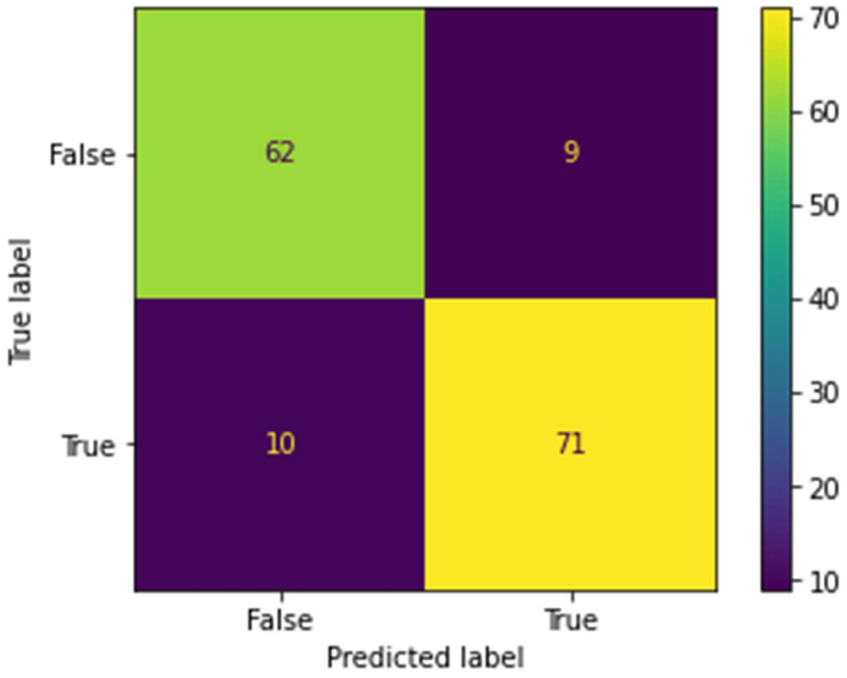
1. Outlier detection



2. Heatmap



3. Confusion Matrix



Conclusion :

We have successfully implemented ML using KNN on the titanic Dataset