Feature Engineering, Transfer Learning (in an Extensive way)

Name –Subham Maity

Roll –14400119043

Subject Code -PEC-CS701E

Subject Name – Machine Learning

Academic Session: 2022-23 (Odd Semester)

Department: CSE(7) College Name: NITMAS (144)

Introduction

What is a feature and why do we need the engineering of it? Basically, all machine learning algorithms use some input data to create outputs. This input data comprise features, which are usually in the form of structured columns. Algorithms require features with some specific characteristics to work properly. Here, the need for **feature engineering** arises. I think feature engineering efforts mainly have two goals:

- 1. Preparing the proper input dataset, compatible with the machine learning algorithm requirements.
- 2. Improving the performance of machine learning models.

The features you use influence more than everything else the result. No algorithm alone, to my knowledge, can supplement the information gained given by correct ***feature engineering****.*

1. *— Luca Massaron*

According to a survey in Forbes, data scientists spend **80%** of their time on **data preparation:**

80%

Time for Data Preparation

20%

This metric is very impressive to show the importance of feature engineering in data science

Imputation

Missing values are one of the most common problems you can encounter when you try to prepare your data for machine learning. The reason for the missing values might be human errors, interruptions in the data flow, privacy concerns, and so on. Whatever the reason, missing values affect the performance of the machine learning models. Some machine learning platforms automatically drop the rows which include missing values in the model training phase and it decreases the model performance because of the reduced training size. On the other hand, most of the algorithms do not accept datasets with missing values and give an error.

The most simple solution to the missing values is to drop the rows or the entire column. There is not an optimum threshold for dropping but you can use **70%** as an example value and try to drop the rows and columns which have missing values higher than this threshold.

Handling Outliers

Before mentioning how outliers can be handled, I want to state that the best way to detect the outliers is to demonstrate the data visually. All other statistical methodologies are open to making mistakes, whereas visualizing the outliers gives a chance to make a decision with high precision. Anyway, I am planning to focus on visualization deeply in another article, and let’s continue with statistical methodologies.

Statistical methodologies are less precise as I mentioned, but on the other hand, they have superiority, they are fast. Here I will list two different ways of handling outliers. These will detect them using **standard deviation**, and **percentiles**.

Log Transform

Logarithm transformation (or log transform) is one of the most commonly used mathematical transformations in feature engineering. What are the benefits of log transform:

Approximate to normal

It helps to handle skewed data and after transformation, the distribution becomes more approximate to normal.

Effect of the outliers

It also decreases the effect of the outliers, due to the normalization of magnitude differences and the model becomes more robust.

Magnitude differences

In most of the cases the magnitude order of the data changes within the range of the data. For instance, the difference between ages **15** and **20** is not equal to the ages **65** and **70**. In terms of years, yes, they are identical, but for all other aspects, **5** years of difference in young ages mean a higher magnitude difference. This type of data comes from a multiplicative process and log transform normalizes the magnitude differences like that.

What is Transfer Learning?

In other words, transfer learning is a machine learning method where we reuse a pre-trained model as the starting point for a model on a new task. To put it simply—a model trained on one task is repurposed on a second, related task as an optimization that allows rapid progress when modeling the second task. By applying transfer learning to a new task, one can achieve significantly higher performance than training with only a small amount of data. Transfer learning is so common that it is rare to train a model for an image or natural language processing-related tasks from scratch. Instead, researchers and data scientists prefer to start from a pre-trained model that already knows how to classify objects and has learned general features like edges, and shapes in images. ImageNet, AlexNet, and Inception are typical examples of models that have the basis of Transfer learning.

Traditional Machine Learning vs Transfer Learning

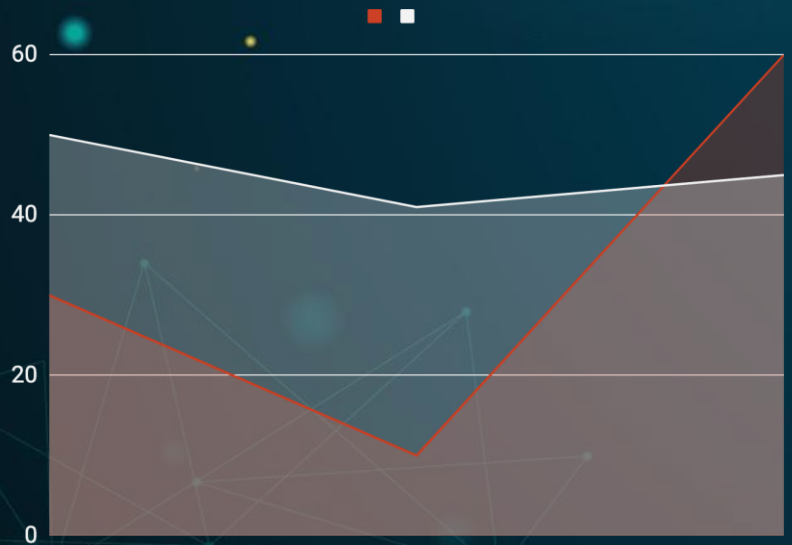
Deep learning experts introduced transfer learning to overcome the limitations of traditional machine learning models. Let's have a look at the differences between the two types of learning.

Traditional machine learning models require training from scratch, which is computationally expensive and requires a large amount of data to achieve high performance. On the other hand, transfer learning is computationally efficient and helps achieve better results using a small data set. 💡 Pro tip: Searching for high-quality data? Check out 65+ Best Free Datasets for Machine Learning

Traditional ML has an isolated training approach where each model is independently trained for a specific purpose, without any dependency on past knowledge. Contrary to that, transfer learning uses knowledge acquired from the pre-trained model to proceed with the task. To paint a better picture of it: One can not use the pre-trained model of ImageNet with biomedical images because ImageNet does not contain images belonging to the biomedical field.

Transfer learning models achieve optimal performance faster than the traditional ML models. It is because the models that leverage knowledge (features, weights, etc.) from previously trained models already understand the features. It makes it faster than training neural networks from scratch.

Transfer Learning vs Deep Learning



Transfer learning: training with relevant and similar data, and realizing the generalization ability of the model itself through transfer learning, How to transfer the learned knowledge from one scene to another.

○ Transfer Learning

● Deep Learning

Key Takeaways

01 What to transfer	02 When to transfer	03 How to transfer
This is the first and the most important step in the whole process. We try to seek answers about which part of the knowledge can be transferred from the source to the target in order to improve the performance of the target task. When trying to answer this question, we try to identify which portion of knowledge is source-specific and what is common between the source and the target.	There can be scenarios where transferring knowledge for the sake of it may make matters worse than improving anything (also known as negative transfer). We should aim at utilizing transfer learning to improve target task performance/results and not degrade them. We need to be careful about when to transfer and when not to.	Once the *what* and *when* have been answered, we can proceed towards identifying ways of actually transferring the knowledge across domains/tasks. This involves changes to existing algorithms and different techniques, which we will cover in later sections of this article. Also, specific case studies are lined up in the end for a better understanding of how to transfer.

Thanks