# Unlocking the Power of Machine Learning

## Address Formatting Correctness Prediction

**Spring 2023**

May 29 2023

## Example - Detecting Typos in Addresses

*Consider the task of detecting typos in street addresses. A machine learning model could be trained to classify addresses as correct or incorrect based on various features, one of which could be the Levenshtein distance from a known correct address.*

## Examples of Addresses

### Correct Addresses

1. 530 5th St E, Northfield, MN, 55057
2. 123 Main St, Los Angeles, CA, 90034
3. 789 Broadway, Apt 12B, New York, NY, 10025
4. 456 Pine Ave, San Francisco, CA, 94108
5. 159 River Rd, Boston, MA, 02115

### Incorrect Addresses

1. 530 5th Street East, North, MN, 55057
2. 123 Mian Street, Los Angeles, CA, 9003
3. 78 Broadway, Apt 12B, New York, NY, 10025
4. 456Pine Avenue, San Francisco, CA, 94108
5. 159 River Rd, Bosotn, MA, 02115

## The Problem

### Incorrect and inconsistently formatted addresses

- *Inconsistently formatted addresses pose a significant problem in data management and analysis.*
- *They can lead to inaccurate data matching, ineffective communication, and inefficiencies in logistics and delivery services.*
- *Incorrect addresses can also pose security issues and lead to lost business opportunities.*

- *Address standardization and validation is crucial in various domains such as e-commerce, logistics, healthcare, and public services.*
- *Data scientists often need to assess the validity of addresses, correct them where possible, or flag potential issues for further investigation.*

Our aim is to develop a machine learning model that can predict if a given address is correctly formatted based on various features.

## One Potential Solution: Random Forest for Address Formatting

### Why Random Forest?

- Ability to handle complex patterns: Addresses can have diverse and complex patterns
- Robust to overfitting: The ensemble nature of Random Forest makes it less likely to overfit compared to simpler models
- Feature importance: Random Forest provides a straightforward way to measure the importance of each feature

### Training Random Forest

- The Random Forest model is trained using our dataset, which contains both correctly and incorrectly formatted addresses.
- Each address is represented by a set of features such as its length, number of words, presence of apartment number, zip code correctness, etc.
- Our model learns to predict the correctness of an address based on these features.

## Feature and Feature Engineering

*"Length"*
*"Number of Words"*
*"State Code"*
*"Zip Code"*
*"Levenshtein Distance"*
*"Presence of apt no"*
*"Presence of street no"*

*"Presence of street name"*
*"Presence of city"*
*"Presence of state"*
*"Presence of zip code"*
*"State code correctness"*
*"Zip code correctness" "Typo in street name"*

## Levenshtein distance

Levenshtein distance is a tool used for quantifying the difference between two sequences by calculating the minimum number of single-character edits needed to transform one string into the other.

- This measure utilizes three types of edits: insertions, deletions, and substitutions.
- Due to its ability to assess string similarity and difference, Levenshtein distance finds extensive use in applications where such determinations are necessary.

Levenshtein, Vladimir I. "Binary codes capable of correcting deletions, insertions, and reversals." In Soviet physics doklady, vol. 10, no. 8, pp. 707-710, 1966.

## Calculation of Levenshtein Distance

Let's consider two strings: kitten and sitting. To transform "kitten" into "sitting" we could:

- ***Substitute "k" with "s" - sitting***
- ***Substitute "e" with "i" - sittin***
- ***Append "g" - sitting***

The Levenshtein distance between "kitten" and "sitting" is therefore 3.

## R code for calculating Levenshtein Distance for Address Validation

```r
# Calculate the Levenshtein distance between the provided and correct address
levenshtein_distance <- stringdist::stringdist(address, correct_address)
```

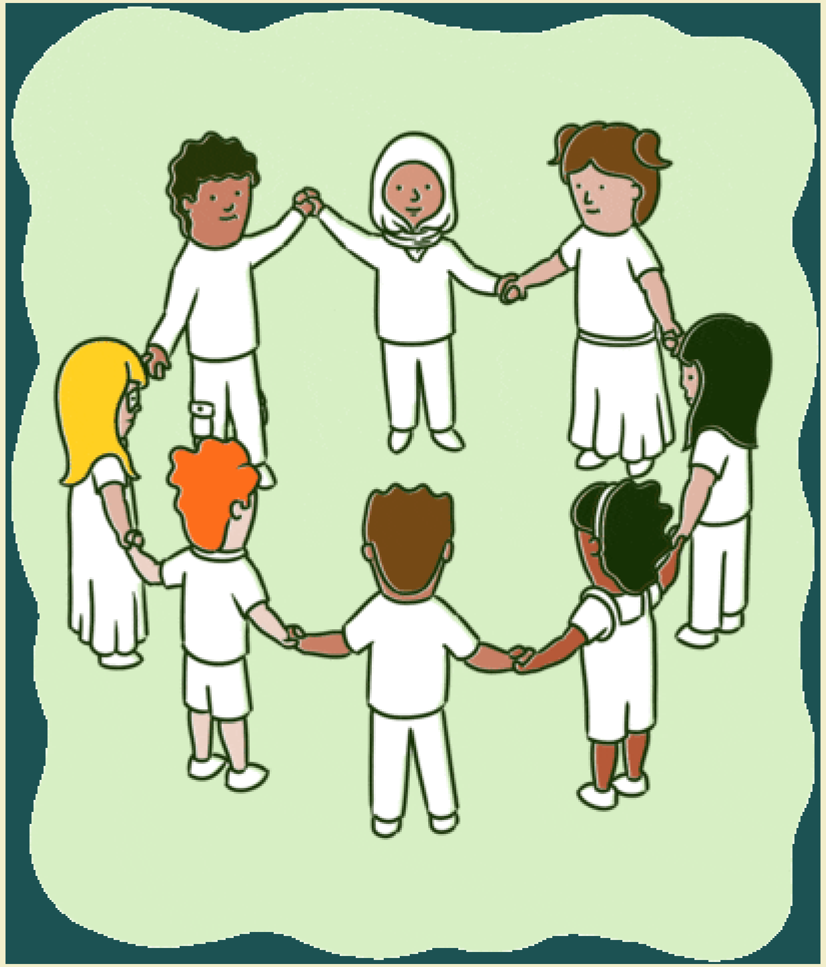
## Other examples

In addition to address validation, the Levenshtein distance can be used in a wide range of other applications.

- **DNA sequence alignment in bioinformatics**
- **Spell checking and correction**
- **Duplicate detection in data cleaning.**

## GROUP ACTIVITY 1

20:00



- Get the class activity 27.Rmd file from moodle
- Let's work on group activity 1 together