# Machine Learning in High Dimension
# IA317
# Sparse Matrices

Thomas Bonald

2023 – 2024

# Sparse data

Data in high dimension are often **sparse** (i.e., have many zeros).

**Examples**

- Textual data (bags of words or $n$-grams)

# Sparse data

Data in high dimension are often **sparse** (i.e., have many zeros).

**Examples**

- Textual data (bags of words or $n$-grams)
- Medical data
- Customer data

| Dataset | #samples | #features | density |
|---|---|---|---|
| MNIST | 10,000 | 784 | $\approx 0.2$ |
| WikiVitals | 10,011 | 37,845 | $\approx 10^{-3}$ |

# Categorical features

| | Account_Balance | Duration_of_Credit_monthly | Payment_Status_of_Previous_Credit | Purpose | Credit_Amount |
|---|---|---|---|---|---|
| **0** | 1 | 18 | 4 | 2 | 1049 |
| **1** | 1 | 9 | 4 | 0 | 2799 |
| **2** | 2 | 12 | 2 | 9 | 841 |
| **3** | 1 | 12 | 4 | 0 | 2122 |
| **4** | 1 | 12 | 4 | 0 | 2171 |
| **...** | ... | ... | ... | ... | ... |
| **995** | 1 | 24 | 2 | 3 | 1987 |
| **996** | 1 | 24 | 2 | 0 | 2303 |
| **997** | 4 | 21 | 4 | 0 | 12680 |
| **998** | 2 | 12 | 2 | 3 | 6468 |
| **999** | 1 | 30 | 2 | 2 | 6350 |

1000 rows × 20 columns

...

**German Credit Dataset**

(mix of numerical features and categorical features)

# One-hot encoding

Using **Pandas**

```
> pd.get_dummies(dataframe, columns=...)
```

| | Account_Balance_1 | Account_Balance_2 | Account_Balance_3 | Account_Balance_4 | Age_years_[0, 10) | Age_years_[10, 20) | Age_years_[20, 30) |
|---|---|---|---|---|---|---|---|
| **0** | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| **1** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| **3** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **995** | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| **996** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **997** | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **998** | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| **999** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

1000 rows × 73 columns

...

**German Credit Dataset**
(encoded)

# Outline

1. **Data structure**
   How to encode a sparse matrix?
2. **Machine learning**
   Which algorithms for sparse data?

# Sparse matrices

$$\begin{bmatrix} 5 & 6 & 9 & 0 & 2 & 2 & 0 & 4 \\ 7 & 0 & 0 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 5 & 5 \\ 5 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 5 & 0 & 0 & 0 & 9 & 0 \end{bmatrix}$$

# Sparse matrices

$$\begin{bmatrix} 5 & 6 & 9 & & 2 & 2 & & 4 \\ 7 & & & & 7 & & & \\ & & 5 & & & & 5 & 5 \\ 5 & & & & & 3 & & \\ 6 & & & & & & & 3 \\ & & 5 & & & & 9 & \end{bmatrix}$$

# Coordinate format

$$\begin{bmatrix} 5 & 6 & 9 & & 2 & 2 & & 4 \\ 7 & & & & 7 & & & \\ & & 5 & & & & 5 & 5 \\ 5 & & & & & 3 & & \\ 6 & & & & & & & 3 \\ & & 5 & & & & 9 & \end{bmatrix}$$

$$\begin{aligned} \text{data} &= (5,6,9,2,2,4,7,7,5,5,5,5,3,6,3,5,9) \\ \text{row} &= (0,0,0,0,0,0,1,1,2,2,2,3,3,4,4,5,5) \\ \text{col} &= (0,1,2,4,5,7,0,4,2,6,7,0,5,0,7,2,6) \end{aligned}$$

# Compressed Sparse Row

$$\begin{bmatrix} 5 & 6 & 9 & & 2 & 2 & & 4 \\ 7 & & & & 7 & & & \\ & & 5 & & & & 5 & 5 \\ 5 & & & & & 3 & & \\ 6 & & & & & & & 3 \\ & & 5 & & & & 9 & \end{bmatrix}$$

$$\begin{aligned} \text{data} &= (5,6,9,2,2,4,7,7,5,5,5,5,3,6,3,5,9) \\ \text{indices} &= (0,1,2,4,5,7,0,4,2,6,7,0,5,0,7,2,6) \\ \text{indptr} &= (0,6,8,11,13,15,17) \end{aligned}$$

# Key properties

The **CSR** format is **memory efficient**

## Operations

Fast...

- **matrix-vector** products
- **arithmetic** operations
- row **slicing**

but slow **updates**

# Outline

1. **Data structure**
   How to encode a sparse matrix?
2. **Machine learning**
   Which algorithms for sparse data?

# Machine learning with sparse data

Example of **scikit-learn**

```
> algo.fit(sparse_matrix)
```

| Algorithm | Sparse data |
|---|---|
| Nearest neighbors | (✓) |
| Dimension reduction | (✓) |
| Ensemble methods | ✓ |
| Naive Bayes | (✓) |
| Sparse regression | ✓ |
| Anomaly detection | ✓ |
| SVM | ✓ |
| Neural networks | ✓ |

✓ Available

(✓) Partially available