Neural Network-Based Text Classification for International Standardized Codes Using R

Johannes Gussenbauer
Johannes.Gussenbauer@statistik.gv.at

Nina Niederhametner
Nina.Niederhametner@statistik.gv.at

Salzburg, 11.07.2024

www.statistik.at

Outline

- Introduction to European Standardized Codes
- Methodological Approach
- Results
- Deployment
- Conclusions and Outlook

European Standardized Codes

European Standardized Codes

- **International Standard Classification of Occupations (ISCO):** Classifies jobs based on tasks and duties
- **Classification of Individual Consumption According to Purpose (COICOP):** Classifies individual consumption expenditures
- **Nomenclature of Economic Activities (NACE):** Classifies economic activities
- **International Standard Classification of Education (ISCED):** Classifies education programs and related qualifications

Task: Assigning **text inputs** from a variety of surveys **to codes** (classes)
→ essential but time-consuming task when done manually

European Standardized Codes
Survey Data

- Data is collected via **household and business surveys**
- Additional information include age, citizenship, education, field of employment
- Codes are highly **imbalanced**
- **Large number of classes** to classify (up to ~700)

European Standardized Codes

Survey Data

Text input (Occupation)	Age	Citizenship	…
Golf Coach	41	AT	…
Kindergarten teacher	34	AT	…
Manager of a Bakery	45	DE	…

Text input (Consumption)	Checkbox	
6 lemons	0	
Chiffon dress	2	
laptop	0	

Survey data

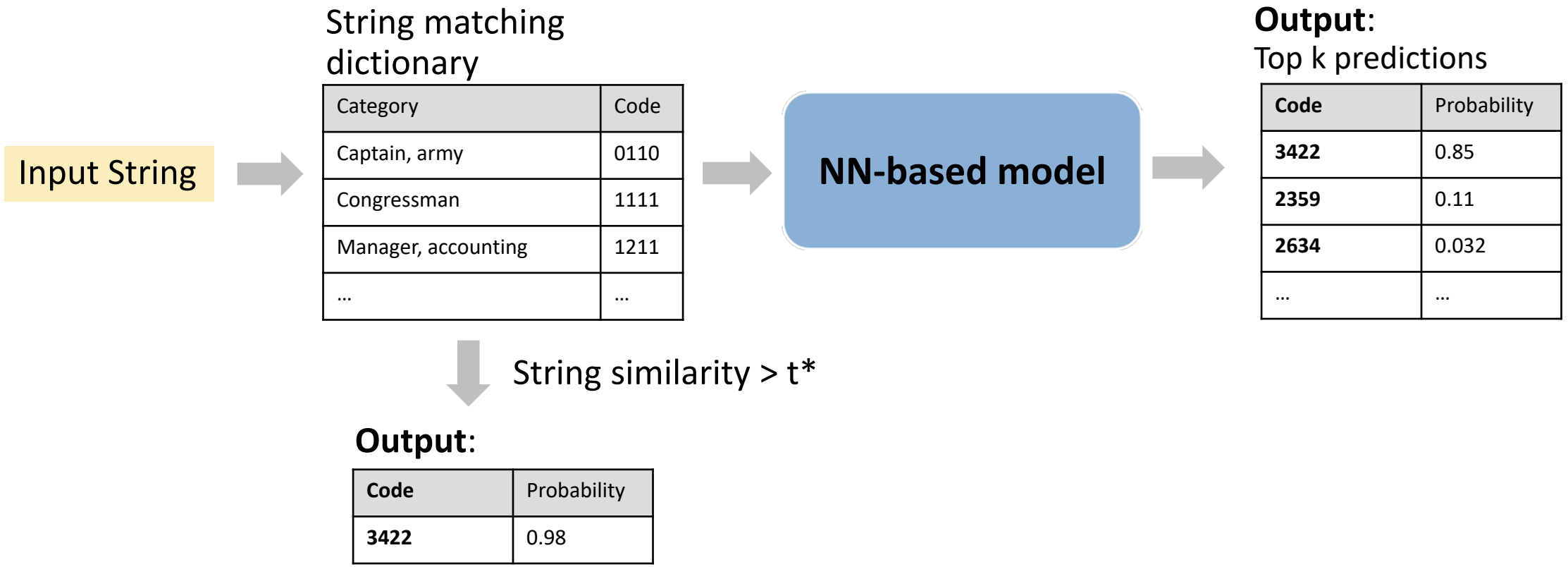
ISCO Code
3422
2342
5223

COICOP Code
01.1.6.2
03.1.2.2
08.1.3.1

Assigned Codes

Methodological approach

Methodological approach
Overview



t*… threshold for string similarity

Methodological approach

String Matching

Input String

"Golf-Coach"

Pre-processing:

"golf coach"

String similarity $\in [0,1]$
with string distance*

Category	Code	similarity
golfer	3421	0.4
coach, sports	3422	0.28
caddie, golf	9621	0.1
trainer, golf	3422	0.08
…	…	

similarity $< t^*$ → NN-based model

similarity $\geq t^*$ → **Output**:
Code | Similarity

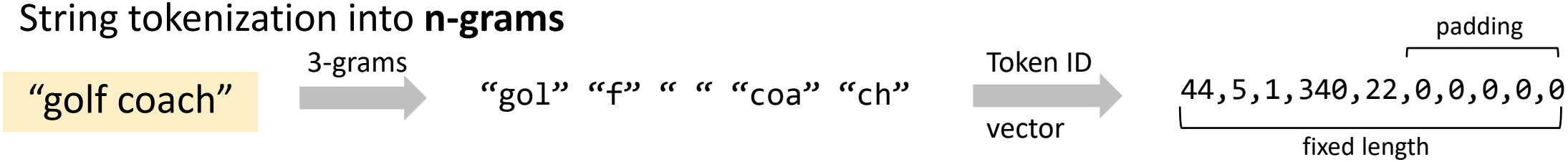
*string distance computed using R package `stringdist`

Methodological approach

Large Language Models

- R packages `keras` and `tensorflow`
  - Recurrent Neural Networks (**LSTM** and **GRU**)
  - **Transformer** Models

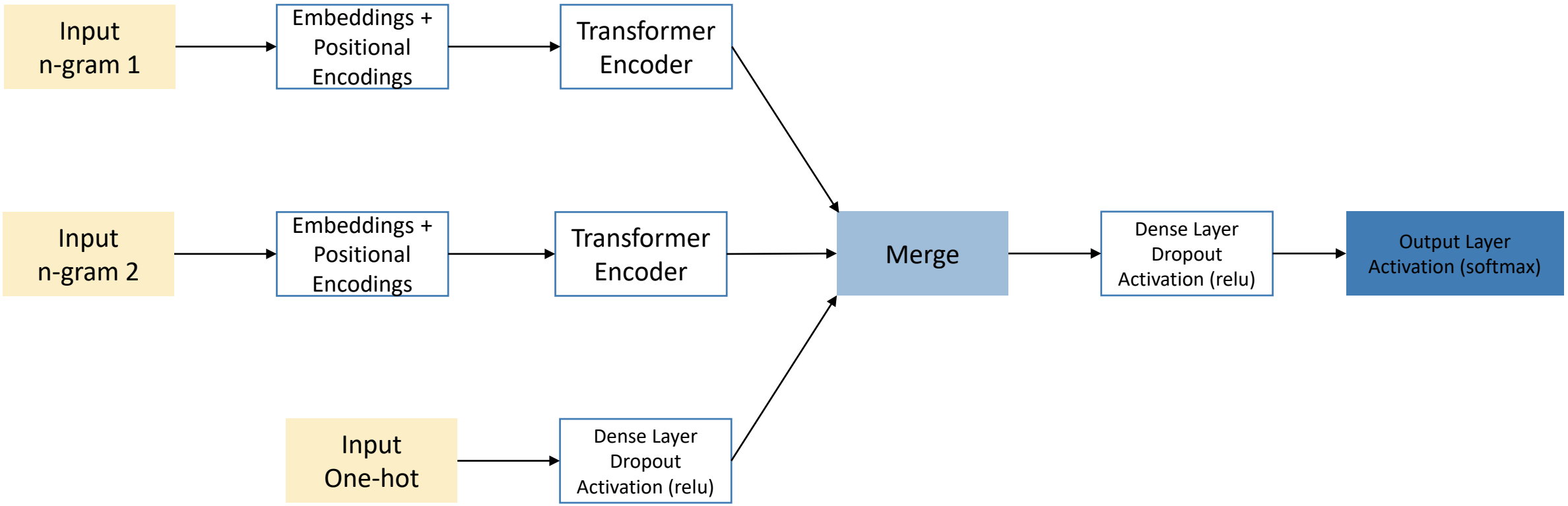
- String tokenization into **n-grams**



- **One hot encoding** for categorical variables and token IDs

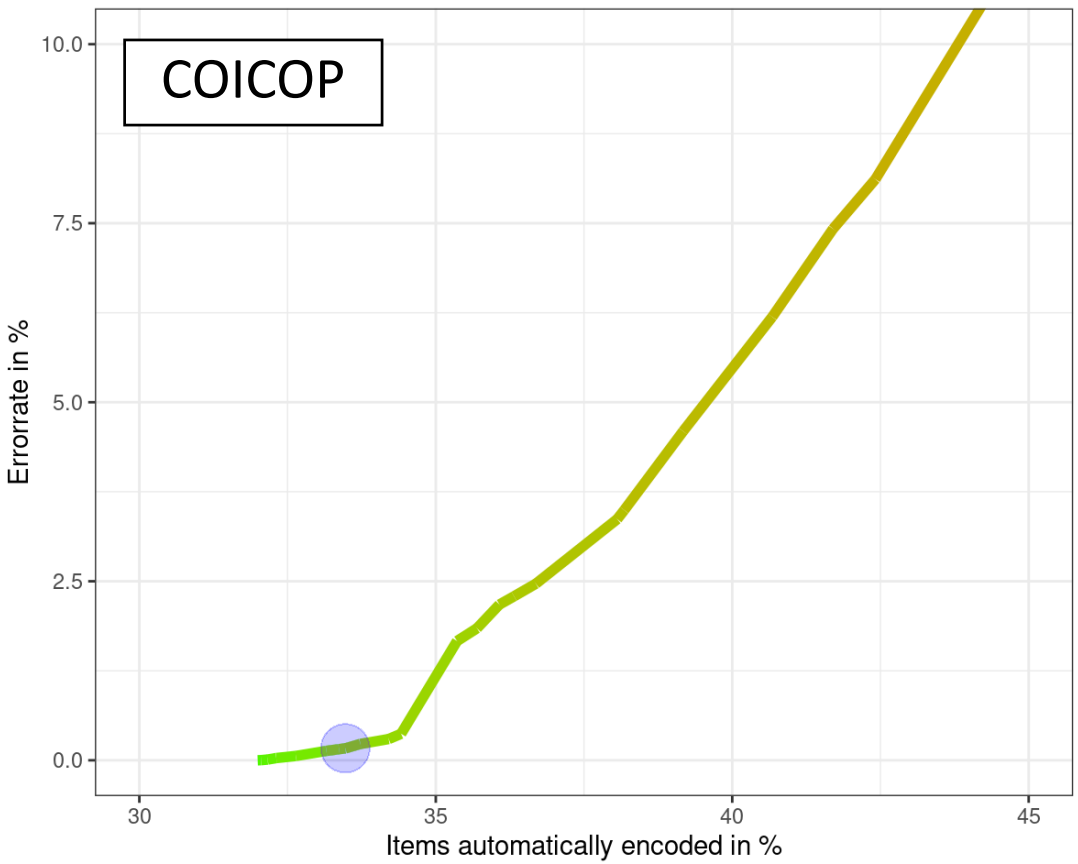
Token0	Token1	Token2	Token3	Token4	Token5	Token6	…	Citizen_AT	Citizen_DE	…
1	1	0	0	0	1	0	…	1	0	…
1	1	0	1	1	0	0	…	0	1	…

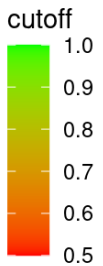
Methodological approach
Example Model Architecture

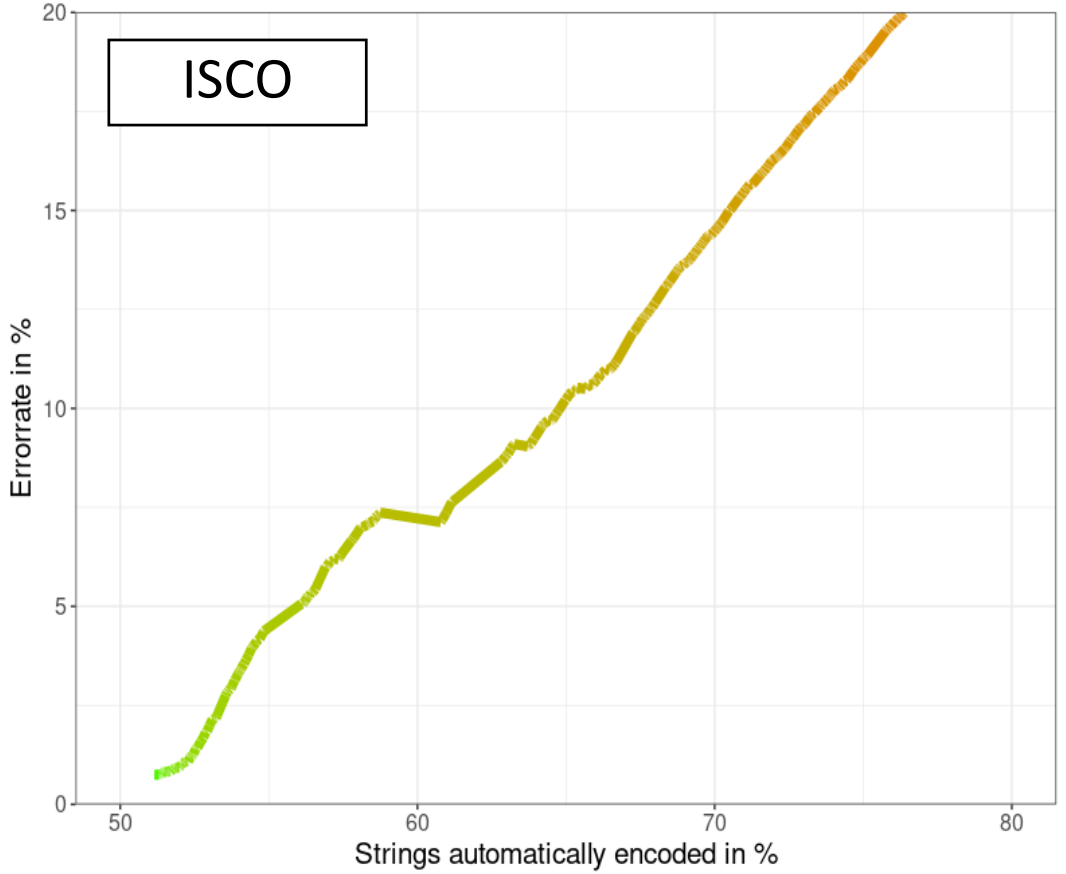


Results

Results
String Matching

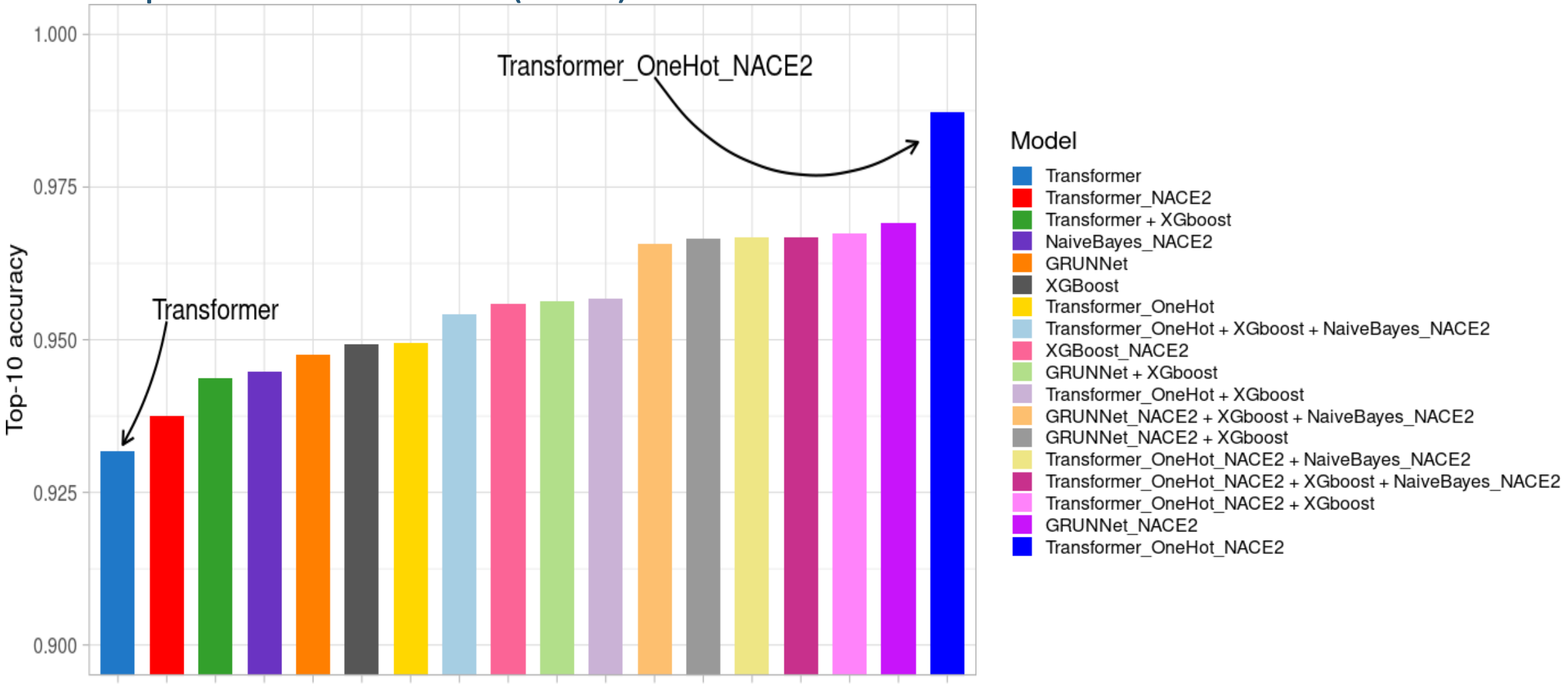






Model Results

Comparison of models (ISCO)



Model Results

Comparison of models (COICOP)

Deployment

Deployment

plumber API

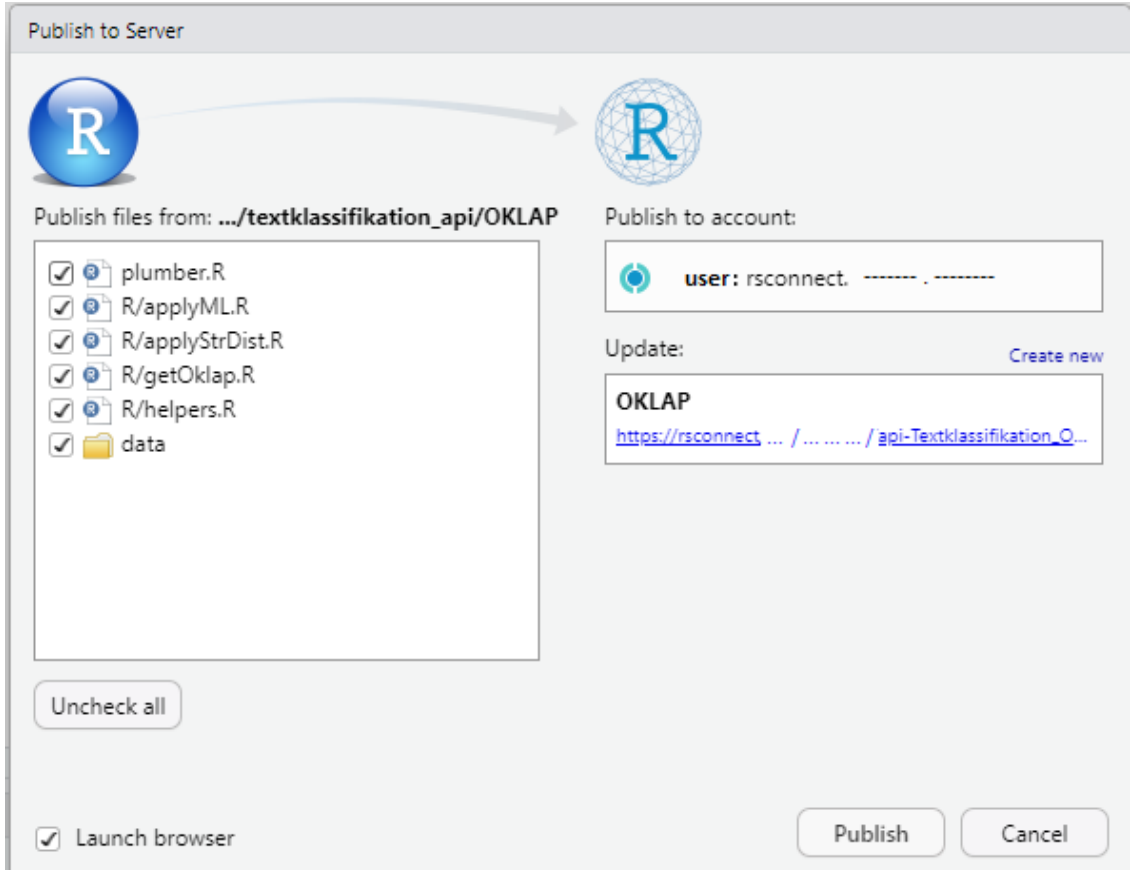
- **Send a request** with input data to the plumber API (json format)
- **Model** makes code **predictions** given input data
- API sends **results** back (json format)
- API is integrated in an App that lets users send requests to the API

- Deploy **one API for each standardized code** due to varying hyperparameters

Deployment

plumber API - publishing

- Hosting on **Posit Connect** integrated into RStudio IDE
- All employees with the API link have access (no access key necessary)
- Predictions are done in **batches**



Deployment

plumber API - requests

REQUEST

REQUEST BODY* application/json

Predictions for ISCO codes

EXAMPLE SCHEMA

```
{
  "top_n": 3,
  "string_dist_cutoff": 0.91,
  "tf_model_cutoff": 0.98,
  "return_probs": true,
  "ordered": true,
  "return_laufnummer": false,
  "input_text": [
    "golf coach"
  ]
}
```

API **request**

Response Status: OK:200
Took 208 milliseconds

RESPONSE RESPONSE HEADERS CURL

```
[
  {
    "text_input": "golf coach",
    "prediction": [
      3422,
      2359,
      2635
    ],
    "probabilities": [
      0.8576,
      0.0297,
      0.0112
    ]
  }
]
```

API **response**

Conclusions and outlook

- LLMs used with top-k predictions work well for our classification use-cases
- Work in progress and potential to improve
- Expand work to other classification codes
- Include Hierarchical Structures into models
- KPIs for API monitoring