CATEGORICAL DATA ENCODING FOR REGRESSION MODELS

Hello,
I am Hristo!

SPO & Power Platform Engineer

- analytics and ML passionate
- museum geek
- in/hthristov
- @HristoChr
- /hristochr

AGENDA

1. Justification
2. Terminology
3. Encoding methods & demo
4. Comparison
5. Future work

1

JUSTIFICATION

Why this topic?

TABULAR DATA 3x4

	Feature 1	Feature 2	Feature 3	Feature 4
Row 1	1	abc	jkl	Stu
Row 2	2	def	mno	Vwx
Row 3	3	ghi	pqr	yz

CHALLENGE

- Feed the tabular data into the regression model
- Text data must be represented numerically
- Simple forms of representation tend to produce poor predictions
- The representation must be numerically consistent among the various features and feature values

CHALLENGE

- Cardinality?
- How to tune my neural network regarding the encoding method?
  - Feature-agnostic results
  - Tuning is a continuous process

2

TERMINOLOGY

Some groundwork

LIST OF TERMS

- Prediction
- Cardinality
- Number of layers
- Number of neurons in each layer
- Hyperparameter (tuning)
- Activation function
- Loss

PREDICTION

- A practical application of neural networks often involves regression
- Predicting quantitative target values explored here is called "regression"
- Regression is a type of supervised learning

CARDINALITY

[...] the cardinality of a set is a measure of the number of elements of the set [...]

TYPES OF LAYERS

- Every neural network has three types of layers:
  - Input
  - Hidden
  - Output

- Typically two hidden layers can approximate the target value with relatively acceptable loss

TYPES OF LAYERS

- quick demo with TF Playground

NUMBER OF NEURONS

- Input layer = feature count
- Hidden layer(s) = hyperparameter
- Output = 1 (regression)

HYPERPARAMETERS

- varying upper and lower bounds
- positive or negative
- integer or floating-point numbers
- categorical

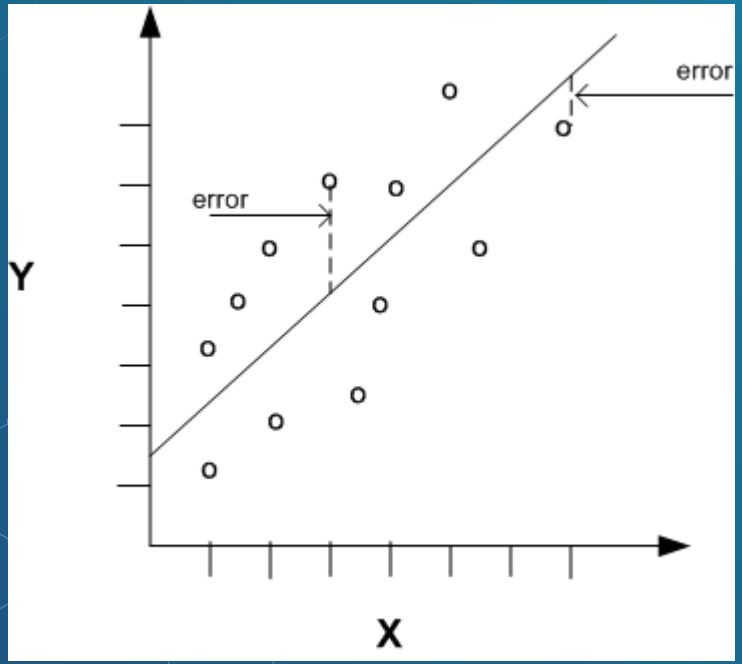
ACTIVATION FUNCTION

- ReLU
  - $(z)=max(0,z)$
- Linear
  - $g(z) = g(z)$

LOSS

- Error functions is what we refer to as “loss”
- Sum of the case-wise error functions for all the cases in a data set
- Regression models rely heavily on the RMSE metric to gauge their performance.

LOSS



- The difference between actual and predicted values

3

ENCODING METHODS

A curated selection

One-hot encoder

For each unique value in each feature, we get a new feature that will contain either a 0 or a 1 for the corresponding case.

ONE-HOT ENCODER

	Caller ID
Case 1	**Caller ID 1**
Case 2	**Caller ID 2**
Case 3	**Caller ID 3**

	Caller ID 1	Caller ID 2	Caller ID 3
Case 1	1	0	0
Case 2	0	1	0
Case 3	0	0	1

Target encoder

Calculate, for each feature value, an average of the corresponding values in the target variable y. Then replace each unique feature value with the according mean.

TARGET ENCODING PROCESS

The mean for the target value is calculated

The count and mean of the target value are calculated

The smoothed mean is calculated

The mean for the target value is calculated over the whole dataset

For each unique feature value, i.e. a grouping is performed per the cardinality

$$\mu = \frac{n \times x + m \times w}{n + m}$$

FORMULA BREAKDOWN $\mu = \frac{n \times x + m \times w}{n + m}$

μ

The mean we are computing. It will be the value that replaces the categorical value c

n

n is the number of values we have per each group

x

x is the estimated mean per group

m

the "weight" you want to assign to the overall mean

w

w is the overall mean.

Binary Encoder

Convert each categorical feature vector into its binary representation after substituting each value with its ordinal representation.

BINARY ENCODING PROCESS

Assign ordinal numbers to the values in the vector

Convert those numbers to their binary representations

Extract the new feature vectors

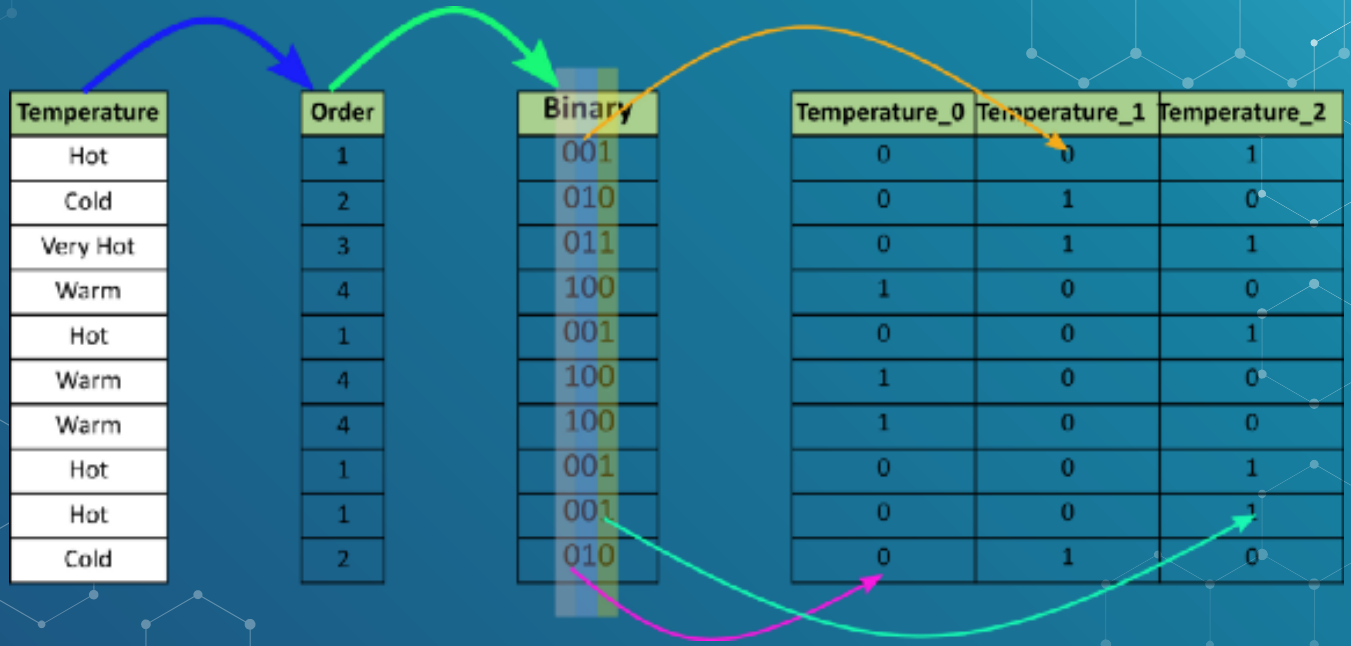
category
category 1
category 2
category 3

category
1
2
3

category
01
10
11

category_0	category_1
0	1
1	0
1	1

BINARY ENCODER



Temperature
Hot
Cold
Very Hot
Warm
Hot
Warm
Warm
Hot
Hot
Cold

Order
1
2
3
4
1
4
4
1
1
2

Binary
001
010
011
100
001
100
100
001
001
010

Temperature_0	Temperature_1	Temperature_2
0	0	1
0	1	0
0	1	1
1	0	0
0	0	1
1	0	0
1	0	0
0	0	1
0	0	1
0	1	0

source

Hashing Encoder

Using a hash function to map feature values to indices in the resulting feature vector. After hashing, we get the location in the vector we have defined by the vector size hyperparameter function.

HASHING ENCODING PROCESS

Take input feature value	Hash the value	Extract the new feature vectors	Assign ordinal numbers to the values in the vector
“Category 55”	Sha256(“Category 55”) = bebba9921075d9a9893029024fb663506ec7ad9a743659ac6fd3fde8e58c6fd7	Convert from the hexadecimal to the integer representation int(hasher.hexdigest(), 16)	int(hasher.hexdigest(), 16) % 15

HASHING ENCODER

	String	Hashed result	Int	Resulting index (%15)
Case 1	**Category 9**	8fb39ab2e53213eeb63d7c75b33cfe7c030d70a794e86f214ccc0ce4fbd75724	6499807066394745246132559455533459825581867195539411996882707408887210779216 4
Case 2	**Category 20**	9e0d81fcaba150a5e640ced7f14155d2ba1f281c0d5f7596c268638af6aad86b	7148929622525826749412618300561880803079436280986926630565401025420875968317 9 14
Case 3	**Category 40**	f324c56964d5ac196dd7fc48a308d79bf1c829ee95bde4c5db3ac460f02d3bea	1099769911855096008399636610683676436143118179997124870045052331799935187179 30 5

Entity Embeddings

Requires a different network topology based on the feature cardinality. One input and one embedding layer are needed for each encodable value.

ENTITY EMBEDDING PROCESS

Establish an embedding size for each categorical feature

Cardinality is input dimension, embedding size is the output dimension.

Each feature is converted into a list of 1 x 85024 values.

int(min(np.ceil((no_of_unique_values_in_feature)/2), 50))

We want to map the larger input space to the more compact output space.

Each value is the index of the unique values.

ENTITY EMBEDDING PROCESS

A multidimensional array is instantiated.

This multidimensional array is passed to model

Its elements are lists containing the encoded categorical values.

From this input, the input layers are extracted and after them the embedding layers. The matrices of the embedding layers are concatenated. The result is a dense layer with an input of size the sum of all embedding sizes.

IMPACT FEATURE EXAMPLE

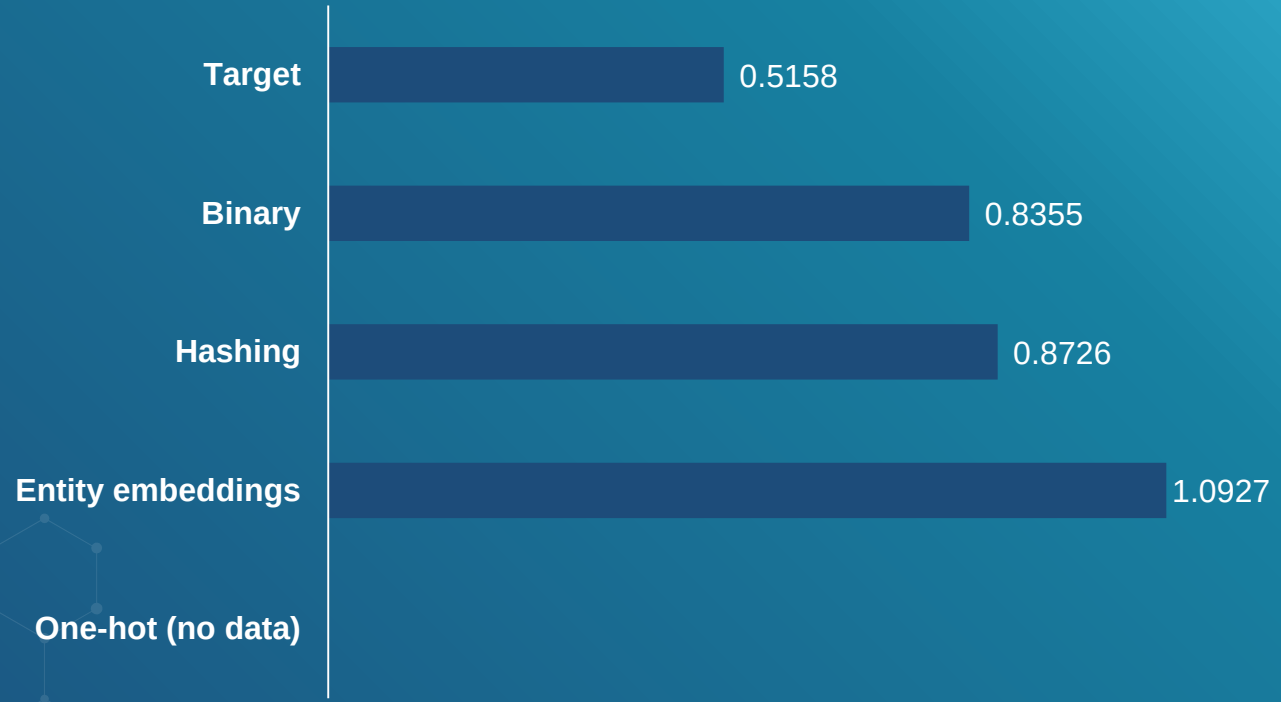
Raw string value	Embedding vector
1-High	[v1, v2]
2-Medium	[v1, v2]
3-Low	[v1, v2]

4 COMPARISON

COMPARATIVE TABLE

	Starting dimensionality	Resulting dimensionality	Performance over 20 epochs, batch size 512 (loss)
One-hot	20	8 010	-
Target	20	20	0.5158
Binary	20	121	0.8355
Hashing	20	20	0.8726
Entity embeddings	20	[15 x emb. size]	1.0927

Loss (less is better)



4 FUTURE WORK

More encoding!

EXPERIMENT YOURSELF

- ml.azure.com (Azure Machine Learning)
  - Create a workspace
  - Attach a compute resource
  - Import data & notebook(s)
- studio.azureml.net (Azure ML Studio)
  - Built-in hashing encoder
  - Run custom python code

WANT MORE?

- Coding Systems For Categorical Variables In Regression Analysis
- All about Categorical Variable Encoding
- Link to notebook

THANKS!

ANY QUESTIONS?

Reach out to me:

- in/hthristov
- @HristoChr
- /hristochr