Categorical variables

The Classification task introduced us to a type of non-numeric variable called Categorical.

A categorical variable

- Has a value drawn from a discrete set called *Categories* or *Classes*
    - The variable that is the target of a Classification task is Categorical
    - Hence the term "Classification" when the target is categorical

There is **no** ordering relationship between category/class values

- { "Red", "Green", "Blue" } (set notation)
- Versus *ordinal* values ["Small", "Medium", "Large"] (sequence notation)

There is no *magnitude* associated with a categorical value

- Even if we could order the colors: how much greater is "Blue" than "Red" ?

We will use C to denote the set of possible values in a category/class.

Since the values in C are unordered, C is mathematically a set of values

$$C = \{c_1, c_2, \ldots,\}$$

Since values in a category/class aren't ordered, they are often non-numeric.

Because computers deal with numbers, we will need to *encode* categorical variables as numbers.

In our Titanic example for Binary Classification, there were two obvious categorical variables

- `Survived` (the target)
- `Sex`

It might have gone unnoticed that the target was categorical

- Because the values were given to us encoded as numeric 1 (Survived) and 0 (not Survived)

We certainly did notice that `Sex` was non-numeric

- Because of it's encoding as text.

Our point is: don't count on the encoding of raw data in order to determine whether a variable is Categorical

We will illustrate this point with the `Pclass` variable, which has three possible distinct values.

How should we encode a Categorical variable with distinct values from a class C where $||C|| > 2$?

An obvious choice for such a variable is to encode the values with distinct integers.

This is usually a **bad** idea !

- ordinal
- there is both an *ordering* and a *magnitude* implied by the encoding

The `Pclass` feature was presented to us encoded as consecutive integers in $\{1,2,3\}$

But it could have just as easily been presented encoded as

- { "First", "Second", "Third" }.
- or $\{1,2,4\}$

Why is the encoding as $\{1,2,3\}$ any more correct than the encoding as $\{1,2,4\}$?

We will give a fuller answer in the module on Model Interpretation.

For now: In a linear model

$$\hat{\mathbf{y}} = \Theta^T \mathbf{x}$$

- Thus, the contribution of the j^{th} feature $\mathbf{x}_j$ to prediction $\hat{\mathbf{y}}$ is $\Theta_j * \mathbf{x}_j$

Consider the encoding of $\text{PClass} \in \{\text{First}, \text{Second}, \text{Third}\}$ as $\{1, 2, 3\}$

- The contribution to $\hat{\mathbf{y}}$ of an example where $\text{PClass} = \text{Third}$ is 3 times greater than when $\text{PClass} = \text{First}$

But had the encoding of $\text{PClass} \in \{\text{First}, \text{Second}, \text{Third}\}$ been $\{1, 2, 4\}$

- the contribution to $\hat{\mathbf{y}}$ would be 4 times greater for an example where $\text{PClass} = \text{Third}$ versus an example where $\text{PClass} = \text{First}$

The arbitrary choice of encoding may have an impact on the prediction.

Bottom line

- Consider whether a feature should be treated as categorical *regardless* of the encoding presented
- Arbitrary mapping of a categorical value to an integer has consequences
    - Avoid it !

We will describe the proper way to encode categorical variables

- And revisit the Titanic example, changing `Pclass` from integer to categorical

One hot encoding (OHE)

So how should we encode a Categorical variable?

If the values come from a class

$$C = \{c_1, c_2, \ldots, c_{||C||}\}$$

then the value can be represented

- with $||C||$ *binary* variables
$$\text{Is}_{c_1}, \text{Is}_{c_2}, \ldots, \text{Is}_{c_{||C||}}$$
- Each is a binary *indicator* variable
- At most one indicator will be true

Here are the possible encodings for each value in C

$$
\begin{array}{cccccc}
 & \mathrm{Is}_{c_1} & \mathrm{Is}_{c_2} & \mathrm{Is}_{c_2} & \dots & \mathrm{Is}_{c_{||C||}} \\
c_1 & 1 & 0 & 0 & & 0 \\
c_2 & 0 & 1 & 0 & & 0 \\
c_3 & 0 & 0 & 1 & & 0 \\
\vdots & & & & & \\
c_{||C||} & 0 & 0 & 0 & \dots & 1
\end{array}
$$

More formally: If the categorical value is c_k

$$
\begin{array}{rcll}
\mathrm{Is}_{c_j} & = & 1 & \text{if } j = k \\
\mathrm{Is}_{c_j} & = & 0 & \text{if } j \neq k
\end{array}
$$

A Categorical variable can be replaced with $||C||$ binary variables $\mathbf{v}_1, \ldots, \mathbf{v}_{||C||}$

- Each an *indicator* or *dummy* variable: $\mathbf{v}_k$ indicates whether the value is c_k or not
    - I like to use the notation Is_{c_k} in place of $\mathbf{v}_k$

This is called the **one hot encoding (OHE)** of a Categorical variable.

- Because at most one indicator in the representation is non-zero

We can use OHE on Categorical variables, whether they be targets or features.

To be concrete: imagine a few rows from our data set

$$\mathbf{X}' = \begin{pmatrix} \mathbf{const} & \mathbf{Sex} & \mathbf{Pclass} \\ 1 & \text{Female} & \text{First} \\ 1 & \text{Female} & \text{Second} \\ 1 & \text{Male} & \text{First} \\ 1 & \text{Male} & \text{Third} \\ \vdots & & \end{pmatrix}$$

After One Hot Encoding:

$$\mathbf{X}'' = \left(\begin{array}{cccccc} \textbf{const} & \textbf{Is}_{\textbf{Female}} & \textbf{Is}_{\textbf{Male}} & \textbf{Is}_{\textbf{First}} & \textbf{Is}_{\textbf{Second}} & \textbf{Is}_{\textbf{Third}} \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ \vdots & & & & & \end{array} \right)$$

OHE can be viewed as a transformation

- which increases the number of features
- A feature from class C is replaced with $||C||$ binary features

Categorical features versus categorical targets

Although OHE can be applied to features $\mathbf{x}$ or targets $\mathbf{y}$, there are some subtle differences in practice

Categorical targets

Although we should use OHE to encode the targets, *in practice* you might see targets encoded as integers

- Binary targets as 0/1
- Other targets such as integers
    - `sklearn` method `LabelEncoder` does exactly this

If it's such a bad idea: why does this happen ?

The answer

- It **may** not matter *from a coding perspective*
    - Often, the code need only be able to *distinguish between* target values
        - e.g., restrict the examples to those with a particular value of the target
    - So the encoding of values is not important
    - In fact: `sklearn` is perfectly happy with non-numeric targets for just this reason !

It **may matter** from a mathematical perspective.

This often manifests itself in the **Loss function**

- where the numerical encoding of the target appears

For example, consider encoding Negative/Positive as 0/1

You might see

- the per-example Loss term expressed differently
- for Positive/Negative cases
- when predicting probability (of example i being Positive) $\hat{\mathbf{y}}^{(\mathbf{i})}$

$$
\begin{aligned}
\mathcal{L}^{(\mathbf{i})} & = & \mathbf{y}^{(\mathbf{i})} * (1 - \hat{\mathbf{y}}^{(\mathbf{i})}) & \quad \text{for Positive example } i \\
\mathcal{L}^{(\mathbf{i})} & = & (1 - \mathbf{y}^{(\mathbf{i})}) * \hat{\mathbf{y}}^{(\mathbf{i})} & \quad \text{for Negative example } i
\end{aligned}
$$

The encoding of Negative/Positive as 0/1 results in

- $\mathcal{L}^{(\mathbf{i})} > 0$
- when $\hat{\mathbf{y}}^{(\mathbf{i})} \neq \hat{\mathbf{y}}^{(\mathbf{i})}$ (mis-prediction)

Categorical features

We would love to make the blanket statement: Always use OHE for categorical features.

Unfortunately, there is one model in which OHE may cause a problem

- Linear Regression, with an intercept
- There is a simple fix (i.e., an argument to pass to implementations of OHE)

The issue is called the *Dummy variable* trap and will be explained in a subsequent module.

Text: another categorial variables

How do you include text variables ? One-hot encoding of the vocabulary !

Example: Spam filtering (Classification task with target: Is Spam/Is Not Spam)

In theory, OHE is the solution

- Vocabulary V of possible words
- $||V||$ indicator variables

In practice

- Vocabulary can be big ! Lots of variables, lots of memory required using OHE
- The representation of a word is "sparse": a single 1 and lots of 0's
    - no relationship between related words: dog, dogs
- Lots of feature engineering possibilities: an ALL CAP feature

We will devote a subsequent module to the topic of Natural Language Processing.

Recap

- We introduced methods to deal with non-numeric variables
- Unfortunately, there are some nuances

In [4]:

```
print("Done")
```

```
Done
```