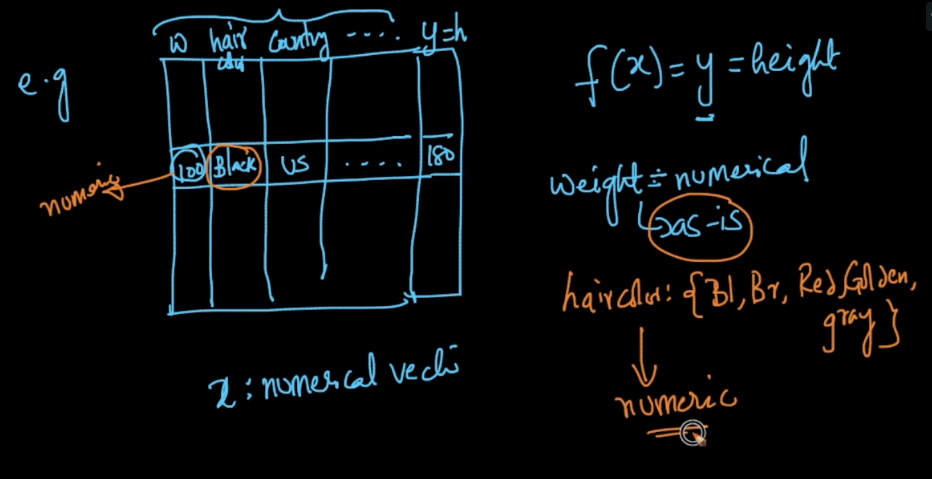
**Let’s say we have categorical feature which is non-numeric so how do we handle that.**

As in given example hair color is a categorical feature with 5 colors, and it’s non-numeric. Now to apply this data to any ML model we need to convert it into numeric value, so we gonna do that.

Let’s discuss each approach one by one.



**1st Approach:**

Let’s say we are given 5 colors, so we assign a number to each color. But numbers are ordinal(inherent order) in nature by default, that means by doing this we are somewhat creating an order between color as

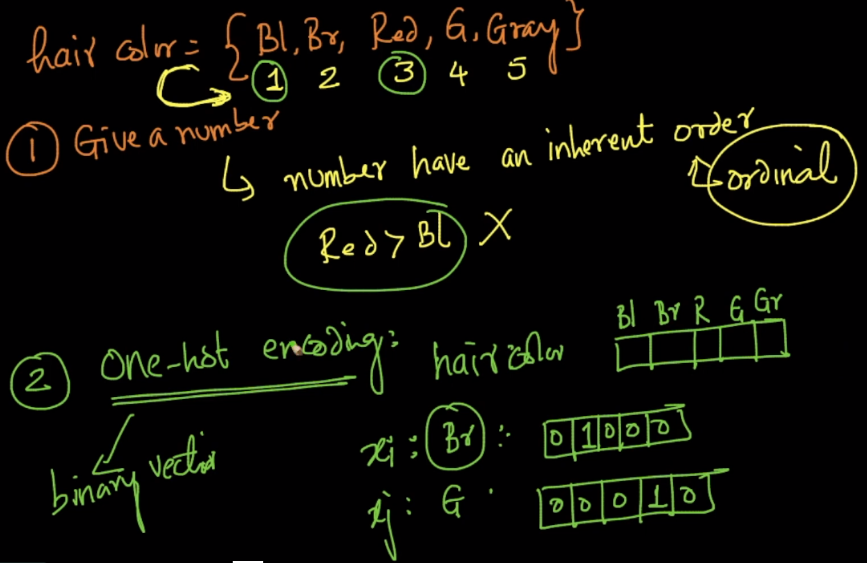
Red > Brown > Black.

But colors are not ordered right. So we’ll not use this.

**2nd Approach: One hot encoding**

Let’s say we have five colors, so what One hot encoding does is that in place of hair color column/feature, it will create five new feature one for each color.

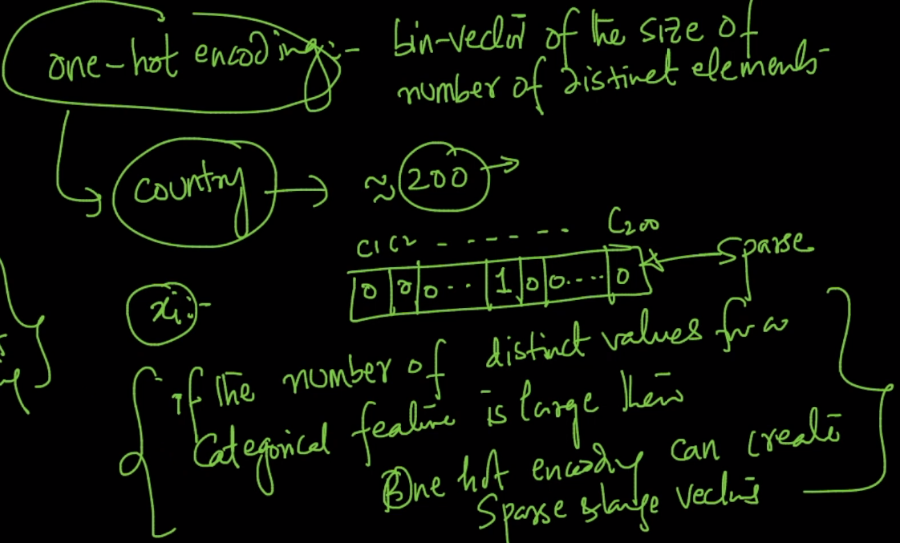
So if any data point have a particular hair color let’s say Red, then value at that Red feature will be 1 and value will be 0 for rest of the hair color features.



**Drawback of one hot encoding:**

Let’s say we have a categorical feature called Country, now in that feature there are 200 different countries, so now if we will do one hot encoding for Country feature, then it will create 200 new features, in which for each data point only one feature will be 1 and rest will be 0.

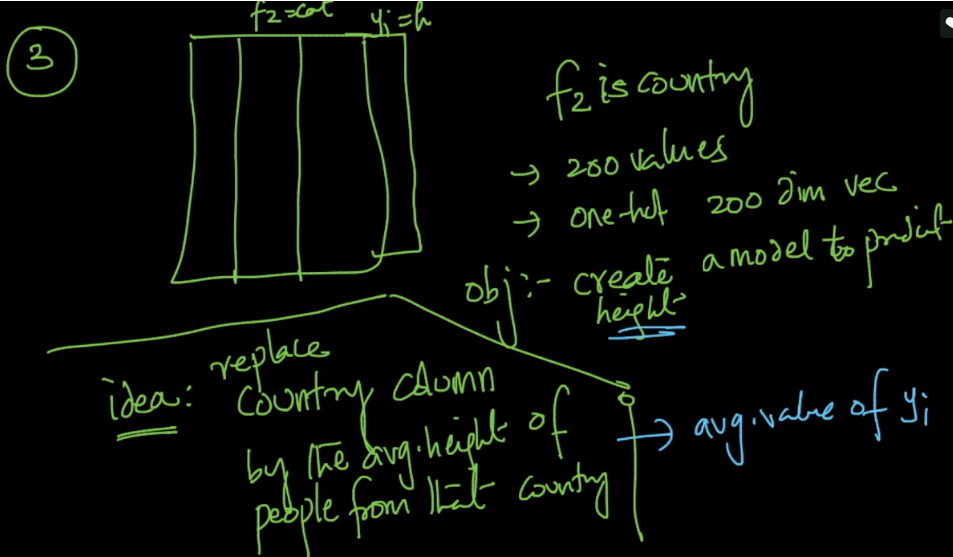
So if number of distinct values for a categorical feature is large then one hot encoding will create a sparse and large vector.



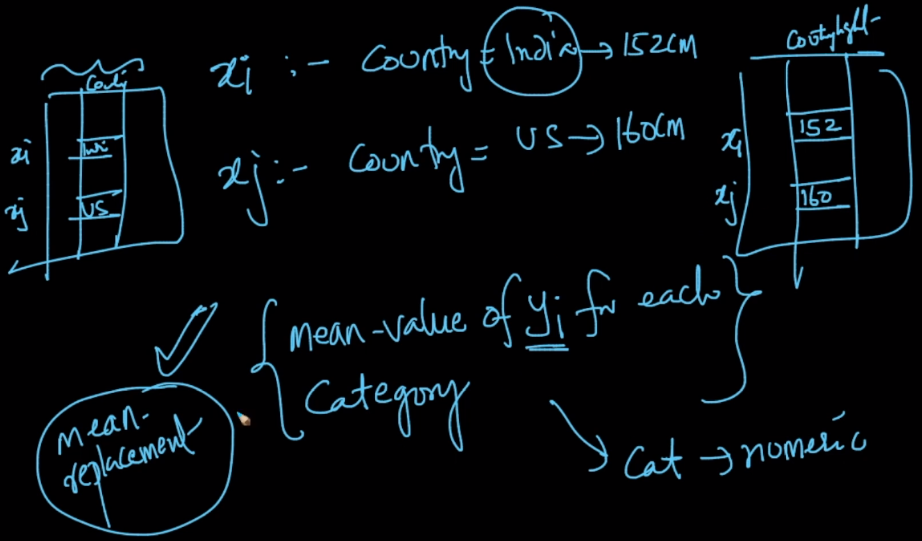
**3rd Approach: Mean-Replacement.**

Let’s say here a categorical feature is Country with 200 distinct country values in that, and our objective is to create a model to predict height. So what we can do in Mean replacement is:

Replace Country column by the average height of people from that country.



Example for each India value we replace 152, and for each US we replace 160.

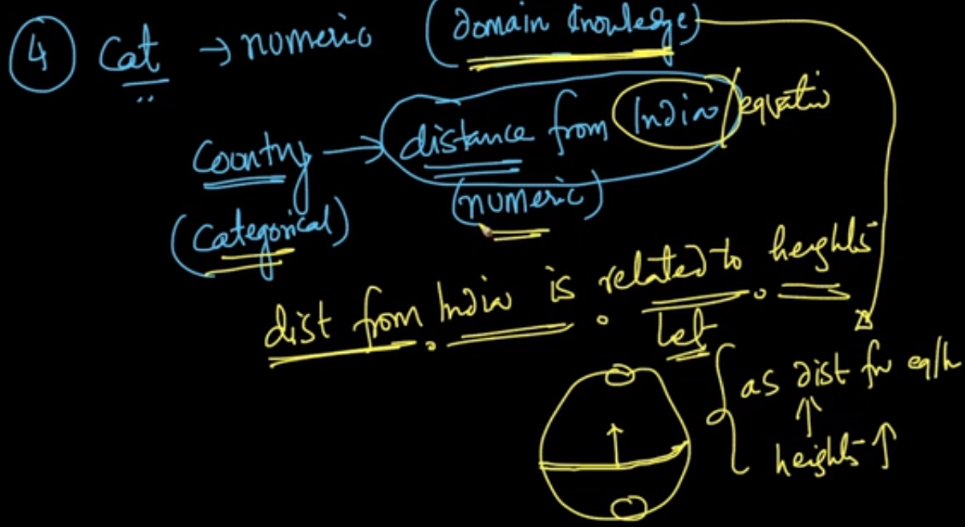


**4th Approach: Convert into numerical using domain knowledge.**

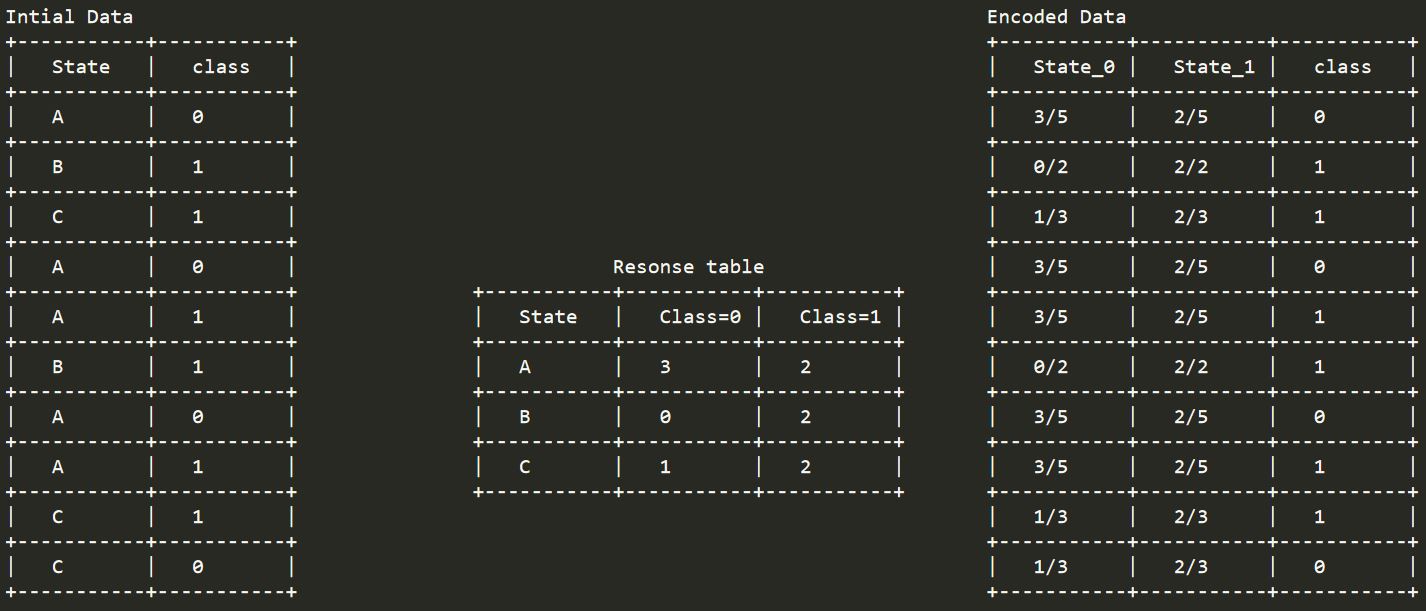
If you have a very good domain knowledge then using the facts that exists in that domain you can replace categorical values with corresponding numeric. As if you go above or lower from equator line it gets more colder.

Or for our ex we replace each country with it’s distance from India.

But for this you must have good domain knowledge.



**5th Approach: Response Coding: It can be used only in classification problems**



We’ll find the no. of values for each o/p for each category, ex state A has 3 values in class 0 and 2 values in class 1, Now we create two columns state\_0 and state\_1,

Wherever we see state A, then in that row and in state\_0 col we place 3/total(3+2) and in state\_1 col we place 2/5

> The response tabel is built only on train dataset.

> For a category which is not there in train data and present in test data, we will encode them with default values

Ex: in our test data if have State: D then we encode it as [0.5, 0.05]

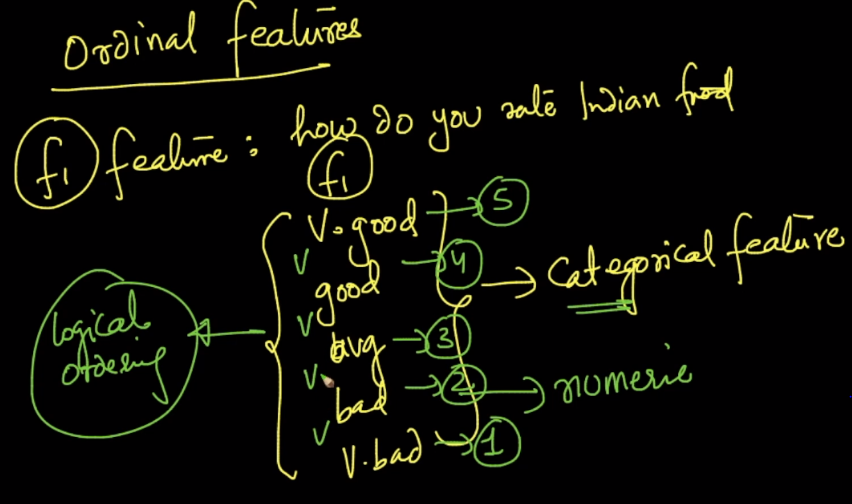
Response coding for binary classification is show below:



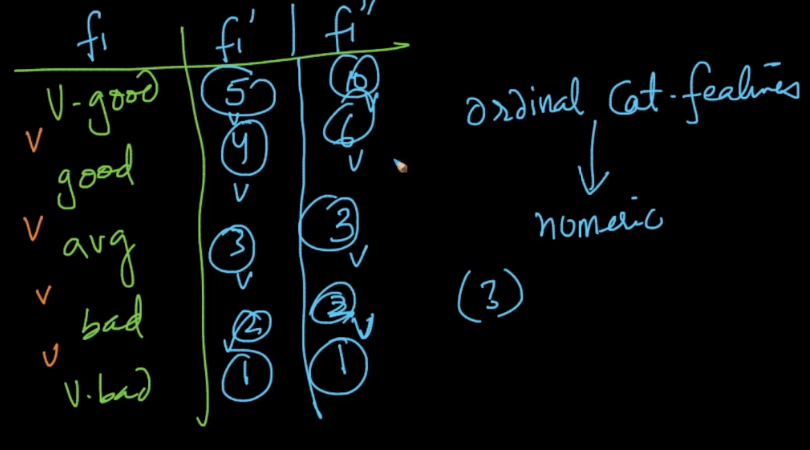
**How to handle Ordinal features:**

Let’s suppose you have a ordinal feature which have a logical ordering on which you can assign numeric value based on ranking.

Example for a feature how do you rate indian food. We can assign rank wise numerical value to each distinct value as 5 for v.good, 4 for good, 3 for avg and so on…

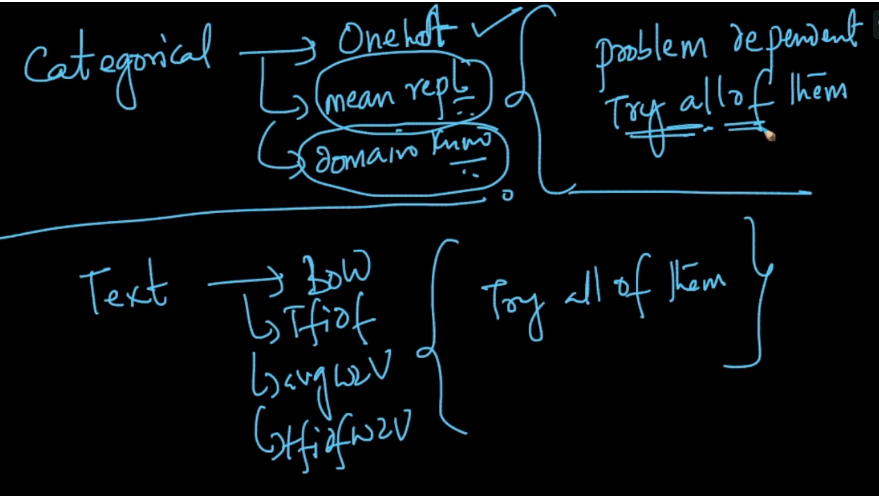


But problem with this what value to assign we can assign 10, 6, 3, 2, 1 here also we are mainintaing order.



**How to know which one to use for particular problem:**

You have to try all of them and use which fits best, as for text we apply all bow, tfidf technique to find best, in this case also we apply all the techniques and then find the best.



Must read link: <https://towardsdatascience.com/smarter-ways-to-encode-categorical-data-for-machine-learning-part-1-of-3-6dca2f71b159>

**Preprocessing using sklearn:** <https://scikit-learn.org/stable/modules/preprocessing.html#scaling-sparse-data>