

~~(Good P - d) - which Data should be binary, pr binary +~~

* Encoding Categorical data (Slide - 7 -)

- If you map categorical data to unique natural numbers, then even if you don't want, you have implicitly ordered the data or assigned rank to it.
- And you can imagine, if you have 20 features (categorical) mapped to natural numbers & you take dot products, you have unintentionally weighted some features more than others due to their high value of natural number.

* Encoding categorical data (Slide - 9 - Hb)

- Imagine you have 'n' features & you decide that each feature must get a unique binary representation so that the mapping is distributed & you don't assign higher values to some & lower values to others^{like} in case of natural number mapping.
- For 'n' features, we would need $\lceil \log_2 n \rceil$ no. of binary bits. & assign ' $\lceil \log_2 n \rceil + 1$ ' extra features for their representation.
- But then, like given in the slide, you would again group some categories together for a feature. Like you would group admin & doctor together for ϕ_{d+2} (see the slide) without it meaning anything. Same for pharmacist & doctor for ϕ_{d+1} .
- So, you need to find an even better way to encode categorical data.

* Encoding Ordinal Data (slide - 16 - 96, 100)

- Ordinal data lies sort of between numeric data & categoric data. i.e. it has numeric data associated with it which is ordered. But the scale may not match with the order. i.e. the differences may not be meaningful.
- An example may be the 'pain scale' that the doctor uses to ask patients for a level or measure of pain. It goes from 0 to 10. But the difference between level 3 pain & level 4 pain may not be same as difference between level 5 & level 6. Because that pain or this scale is not quantified properly. It doesn't have any unit of measurement or scientifically defined formula for measurement.
- So, you can't treat ordinal data like numeric or categoric. You wanna get a new way of encoding it.
- One way could be ~~the~~ unary / thermometer code - Like for 'n' different levels of ordinal data, you take 'n' binary bits i.e. 'n' features with binary values. And you fill it up with 1s as you go up a level (like shown in the slide).

Encode Numerical Data

(Standardize Numerical Data)

(slide - 21, 22 - 113, 114)

- If you have two features with different scales like one with values ranging in 10,000s & one with values ranging in ~~100s~~ 100s, then you will have

(25 - abir) ~~understanding standard deviation~~

issues visualizing data & interpreting it from visualisation.

- Standardization distributes the data around zero & changes its scale. This gives much better results.
- In standardization, you take a ^(feature) data point, you subtract its mean from it & you divide ~~the data~~ that by the standard deviation.

★ More benefits of plotting your data (slide 23 - 18) (& talking to experts)

- It's always beneficial to visualize your data. It may give you an idea about how ~~to~~ the data is, how is one feature related to other, etc.
- You will be able to improve your model from good interpretations of your visualization.

NOTE:- What are quantiles? (Question)

* Classification Boundaries (Slide - 27 - 147) (3-D functions)

- It's better to visualize these classifiers in 3-D (for 2 feature vectors), where the 3rd dimension \vec{z} is defined by hypothesis function.
- For example:- Like given in the slide, for linear classifiers, imagine $\theta^T x + \theta_0 = z$, where when $z=0$, we have the classifying hyperplane.
In case of non-linear concentric boundaries, you can define $z = f(x)$. In both cases, labels are assigned for $z > 0$ & $z < 0$. (+1 for $z > 0$ & -1 for $z < 0$)

* Non linear Boundaries:- (slide - 28 -)

(Taylor expansion example)

- You really have to decide yourself on how to choose 'z'. It could be a linear classifier (two planes intersecting each other) with linear boundaries, you can set it to polynomial functions like $z = f(x)$ where $f(x)$ is a polynomial in x_1, x_2 .
- To determine what & how to choose you must really visualize the data.
- In this case (slide) you expand the feature space by using taylor expansion on both x_1 & x_2 .

* Non linear Boundaries (slide - 29 - 167)

- You can see in the slide, that for a dataset that doesn't obviously seem classifiable, we can classify it just by creating a smooth function in 3-D (for 2 feature-vectors)
- Thus, you can linearly classify ~~≠~~ data even for non linear boundaries.
- But, this also means that with enough computational & mathematical power you can linearly classify extremely cluttered & complex non-linear boundary data.

nonlinearly fit with smooth (robust method) is not

* Nonlinear, Boundaries (slide = 30 - 174)
(Heart rate vs. Exercise Example)

- (Continued from previous slide) So, although you get a classifier for overlapping cluttered data with training error

(
- S-ability) words in not French n'importe + it's not
as zero, it may be overfitting. You may have perfected
the classifier for the training set, but it may fail on new
sets for prediction.

For example, you may so overfit data for resting heart rate
vs. exercise time that even 0.01 change in any parameter
changes the label. Like you may get heart disease for
78 bpm & no disease for 79 bpm & again disease for
80 bpm. That seems illogical. It means you have just
overfitted your data.

★ Evaluation of Learning Algorithm : (Slide - 31 -)

- One good solution is to train your classifier only on one portion of your data. So that you can test it on the remaining data.
- How much portion to choose depends on you. More training portion means high chances of overfitting & low amount of data to test on. Less training portion means your classifier will obviously be weak. It's a compromise.
- Now, since you are dividing your data into portions, make sure you shuffle it first. You don't want your portions to be biased. (For example:- You are collecting data from hospitals & your training portion ~~contains~~ has majority data from only one hospital. There will be bias)