Lecture 8 - mud cards

October 4, 2021

1 Mud card questions

1.1 General questions/concerns

- **In general I find the course is outpacing me- the concepts are increasingly abstract/based on ideas we've only recently covered- I think the homework might help with this if it has several of the exercises like the quizzes.**
  - Unfortunately the concepts will only get more complex
- **I definitely like in person class much more. I feel that I am more involved and can interact more which leads to better retention.**
  - yep, definitely agree
  - teaching (and learning) last year was tough
- **I noticed transformed train features: [[0. 0. 0. ... 1. 0. 0.]] that there is a dot of every entry, so I mean is this a string or integer? And how do use this matrix in Machine learning?**
  - that's just how python prints out matrices that are too large.
- **Can you explain sparse matrices again? Still a little bit confused by the subject!**
  - a matrix is sparse if it contains lot of 0s
  - in that case, the matrix can be represented in a more memory-efficient way

1.2 Preprocessing

- **When we do the encoding, when should we have one less dummy variables?**
  - `drop{'first', 'if_binary'} or a array-like of shape (n_features,), default=None Specifies a methodology to use to drop one of the categories per feature. This is useful in situations where perfectly collinear features cause problems, such as when feeding the resulting data into a neural network or an unregularized regression.`
- **The muddiest part for me is to use onehotencoder or ordinalencoder. Because some category can't be ranked, but as mentioned in class, it might be followed into a time line.**
  - if you are referring to the marital status feature, that's wrong
  - use common sense and social sensibility
- **So why does the output for unordered categorical data consist of only 0 and 1..? I expected there will be more numbers based on various pair of variables.**
  - one point only belongs to one of the features
  - I'm not sure what you mean by pairs
- **I am confused why someone would choose to randomly shuffle versus iterating through, the later seems more rigorous and just better.**

  - maybe but it can be computationally prohibitive to iterate through every possibility (e.g., if you have thousands of groups)
  - sometimes it's better to randomly shuffle
- **"I still don't understand why the future information in the validation/test set has to be immediately following the data in the training data. I understand that you shouldn't be using future data to predict past data, but what is wrong with using data on, say, Monday to predict something for Thursday, or rather using data on Thursday to validate predictions on data from Monday?**
  - yep, that works as long as you don't use any data from the in-between days
- **Why do we do fit before transform for training data?**
  - try to transform first without fitting :)
  - you will get an error message
  - transform uses the results of .fit (e.g., the mean and std of each feature) to do the transformation
- **When we choose which encoder to preprocess the data, is that also depends on what we want to achieve?**
  - usually not
  - the way you preprocess a feature should not depend on your target variable
- **I am still confused about the ordinal encoder. For example, if we assign 5 to excellent and 1 to poor, does it mean that excellent is five times the value of poor? Will it add bias to our model?**
  - it might
  - if you can assign numerical values to your categories, you should not use the ordinal encoder
  - you should replace each category with the corresponding value
- **What are some scenarios that we should use GroupSplit?**
  - any time there is group structure in your dataset
  - e.g., a patient/customer/any sort of entity is described by multiple data points
- **What do we do to the outliers in the val/ test set after fitting minmax scaler on training set?**
  - nothing
  - if training set min max after transformation will be 0 and 1
  - the validation and test sets might not be and that's fine
- **How does the StandardScaler handle missing values in a dataset? Or does the transform() simply do nothing with NaN values?**
  - Create a simple dummy dataset and try it
  - I'm not 100% sure what happens

1.3 Missing values

- **I'm confused about Quiz 3. Can SimpleImputer function use for the non numeric array?**
  - yes if you use the most frequent or constant strategies
  - let's check quiz 3
- **For continuous missing data, (multivariate) imputation is not good, but we still use them?**
  - my advice is that you should not use it
  - there are better approaches we will cover in November

  - but people use it, your boss might insist that you use it, so you should be familiar with it
- For the several techniques dealing with missing data, we have to deal with missing data based on another column with no missing data, how to start if all column contain some missing dataset?
  - multivariate imputation works if all features contain some missing values
  - the advanced techniques we will cover in November also work
- **I thought that the muddiest part of this section of the course was understanding when to use MCAR, MNAR, or MAR.**
  - these are just concepts trying to make sense why values are missing from a dataset
  - it's not osmething you'll need to use in practice
- **For SimpleImputer for filling in for ordinal variables: where does the ""fill_value"" of ""NA"" go in the ranking? Does it go to the lowest rank, or the highest? Eg. Letter Scores A, B, C, NA; what does this look like in the encoding?**
  - that's feature-dependent, use common sense
  - if the feature describes grades at Brown, the order would be NA, C, B, A because NA means you didn't get a grade
- **How are we estimating the uncertainty with multiple imputed datasets with different random states? Are we taking the mean and stddev for each predicted target variable in the test data? How do we aggregate these uncertainties across the whole test set into an overall measure of uncertainty?**
  - the standard deviation is your measure of uncertainty
  - the larger std is, the more your model performance depends on the random state and the more uncertain your model is
- **"From the wikipedia https://en.wikipedia.org/wiki/Missing_data"Missing at random (MAR) occurs when the missingness is not random..." I can't be the only one that finds this confusing**
  - nope, it is super confusing!
  - MAR should be called missing conditionally at random but that acronym is also MCAR which is taken :)
- **Should we just delete a column if it has way too much missing value?**
  - I'd advise to not delete it

1.4 Feature selection

- **How can we determine whether the interaction terms work out well in real-world practice as there are multiple factors influencing the performance of the model?**
  - you can measure feature importance using the feature selection tools we covered
  - you can do an experiment:
    - train a model using the original feature and measure the validation score
    - train a model with one new feature using the exact same split as before and measure the validation score
    - if the validation score improves, the feature is predictive
  - make sure to change one aspect at a time
    - if you add multiple new features and the socre improves, you won't know which feature is the reason for the imporvement!
- **If you have some features you want to use both the F test on and some to use**

Mutual Information on, how do you select the K best?
  - you could aggregate the results of multiple selectors
- **The difference between SelectKBest, and SelectPercentile is it same or different?**
  - they do the same calculation but return features in different ways
  - SelectKBest return the K best features
  - SelectPercentile return the X percentile best features
- **I am little confused about how to determine the features we are going to include. Do we need to loop over every possible combination of the features, and plot the test_score? Or do you just need to select certain features that may make sense, and add some or drop some to see if the test_score changes?
  - always use all features if you can
  - feature selection should only be used if the number of features greatly outnumbers the number of datapoints
  - never drop a feature based on intuition alone!
  - make data-driven decisions
- **Is there feature selection method that takes feature interactions into account?**
  - not in sklearn as far as I know but you can write one
- **is there a way to determine optimal number of features/stopping point where adding more is not adding significant improvement or is that something you have to decide yourself?
  - experiment
  - add one feature at a time and measure the best validation score
  - keep in mind that there is no correlation between the number of features you add and the model improvement

1.5 Feature engineering

- **What does bias = 1 mean in automatic feature engineering?**
- **confused about how to find the way to do Manual Feature Engineering**
  - it's tough because it is dataset-dependent
  - there is no approach that works for all datasets
  - think about the target variable, consider what feature could be predictive
  - read papers, talk to your team and subject matter experts
- "For the last quiz, why is the shape (9,3) instead of (3,9).
  - yep, you are right, it should be (3,9), three points and 9 features.
- **Why do we use polynomial to transform the features!**
  - it's a simple transformation and it might work well sometimes
- **what are the different ways to engineer a new non linear feature?**
  - you can apply any non-linear function to a feature
- **I would like to confirm that if we decide on imputing the missing values, we still need to do it after data split
  - yes!
  - the iterative imputer has the same .fit and .transform methods as any transformer

- **Is PolynomialFeatures method the only automatic feature engineering method in sklearn? What if there are situations that it cannot solve? In that case, the only choice is to do the feature engineering manually, right?
  - here is a list of all automatic feautre engineering methods in sklearn
  - yes, manual feature engineering is usually preferred and you'll likely spend a lot of time on this
- **Do you recommend that we try to do manual feature engineering for our midterm project?**
  - YES!
  - it's not a requirement because it's tough to do
  - but if you have time, do it
- **How should we determine whether we should set `interaction_only` to True or False in automatic feature engineering?
  - try both and do an experiment do determine which one gives you a better model performance
- **If we run either an f-regression or mutual-info-regression and don't get great correlations with either, should we just attempt feature engineering if we feel we have enough domain knowledge?
  - yes
  - also consider collecting more data if that's feasable

[ ]: