1. With feature selection RFE, is it normal that the R squared decreases? Because we explain the dependent variable with less features although these features are very strong. From what I have understood, it is normal that R-square should be lower but if it is not so much lower from the initial R-square with whole features I guess it is fine to prove the power of these important features

Yes, correct! It is normal that the R squared **decreases slightly** in RFE - typically, RFE would kick out the features which do not add much explanatory power (but can still add a little bit), so it will decrease by some amount. Ideally, the remaining features will carry the majority of the explanatory power so it will not decrease by a lot.There is a resource which goes through a whole modelling process also talks about RFE and decreases in R squared, and also mentions something that we will take a look at tonight, which is **cross-validation**.<https://medium.com/@feraguilari/multiple-regression-and-recursive-feature-elimination-rfe-34af0c6ae51b>2. When we use OneHotEncoder, should we drop any columns during concatenation like in get\_dummies or we are fine with using all the columns? Similarly to pd.get\_dummies, OneHotEncoder has an argument, which allows you to drop\_first - for the OneHotEncoder, the argument is called **drop**:  
drop{'first', 'if\_binary'} or a array-like of shape (n\_features,), default=NoneSo the **choice of function** which we choose to generate the dummies (pd.get\_dummies or OneHotEncoder) does not have an impact on dropping the first category, since both allow for it.There is a separate discussion, however, which concerns for **which ML algorithm** you need to or do not need to drop the one category for categorical variables. We always need to drop\_first for linear regression and logistic regression, since otherwise we get a problem of multicollinearity. There are other algorithms, which are good at handling non-linear dependencies and also do not have a problem with correlated variables, namely DecisionTrees and SVM of the ones we have seen (on a side note: in the mushrooms dataset, that you used for the last lab, the variables were correlated to each other, which is why the lab instructions suggested using a DecisionTree and SVM). So technically, when using those, we do not need to drop the first category.Here’s a discussion on StackOverflow about this also: <https://datascience.stackexchange.com/questions/47638/in-which-cases-shouldnt-we-drop-the-first-level-of-categorical-variables>3. Do we want to also standardise the dependent variable, does it ever make sense?Typically, when doing feature scaling, we are interested in standardising the **features**, so the independent variables / predictors. The reason for this being that especially distances-based algorithms will do poorly when features are on different scales and might put more weight on some of them. That said, we typically don’t standardise the dependent variable. If we are interested in the prediction, that what we would predict is the **standardised Y** instead of the Y - which is something we cannot easily interpret, would have to transform back again etc. It’s not exactly related to the question, but here is also another resource which has a nice overview over supervised ML algorithms including which ones of them require feature scaling:<https://www.dataschool.io/comparing-supervised-learning-algorithms/>

[Multiple Regression and Recursive Feature Elimination (RFE) | by Fernando Aguilar | Medium](https://medium.com/@feraguilari/multiple-regression-and-recursive-feature-elimination-rfe-34af0c6ae51b)

[machine learning - In which cases shouldn't we drop the first level of categorical variables? - Data Science Stack Exchange](https://datascience.stackexchange.com/questions/47638/in-which-cases-shouldnt-we-drop-the-first-level-of-categorical-variables)

[Comparing supervised learning algorithms (dataschool.io)](https://www.dataschool.io/comparing-supervised-learning-algorithms/)