* Project Goals
  + What is your goal ?
  + What is the data about ? Where is it from ?
  + How will you approach the problem ?
* Exploratory Data Analysis
  + Is imputation needed i.e. are there NA’s ? If not still show your work. If yes clean the data.
  + Are there duplicates ? Should you clean them ? (DataFrame.duplicated() and DataFrame.drop\_duplicates() functions can help). We will discuss the effect of having duplicates
  + Discuss the input features. Which ones are categorical ? Which ones are not ?
  + Do you need to one hot encode the categoricals ? That will depend on the eventual algorithm you choose. For example any tree based algorithm (RandomForest, Gradient Boosters etc) can handle categoricals without requiring encoding. Other models will require encoding. Think about do you want to just pick one algorithm and stick with it ? Or do you want to do a few and compare ? No right or wrong way here.
  + Do correlation analysis between the input features and the label. You can use the corr() function on the dataframe. Draw correlation plots to show the work. Seaborn has some nice plots. What does the correlation tell you ? Are there input’s that have no correlation with the label ?
  + Is the data balanced i.e. do you have similar number of samples in all the label categories ? The Counter class or np.bincount() will come in handy here. If there is a lack of balance what does this mean ? We will talk about this in more details as well.
* Feature Selection
  + Are all features equally important ? The correlation work may give you a hint, but not the best way to select.
  + Look at feature selection in the sci-kit learn docs. We will talk about this in more details – but basically you can do chisquared tests (or f tests) to see which ones have a p-value less than 0.05 and pick those i.e. pick the significant ones.
* Modelling
  + Splitting the data. Think about what a right kind of split is. The dataset is pretty large – so chances are you do not need a large test percentage. Save your test data as a separate csv file – may come in handy later.
  + Are you going to compare multiple models ? Or just stick with one model ? In either case you should do a baseline training on the model(or models) you want to work with along with some hyper parameter searching and cross val. Your goal should be to see if you can then improve on the best validation score.
  + Choosing the right metric is very important. Think what might be the best scoring metric here. Is accuracy a good metric ? Or should you consider precision/recall ? What about both (via the F1 score) ? Or may be AUC score ? Will talk through this as well.
  + Check the returned scores to ensure you are not overfitting. If you are overfitting – try to regularize.
  + If you are comparing multiple models, you might want to pick the model with the best baseline score and stick with it – or try to improve all of them. Again no right or wrong answer – depends on how much time you have.
  + What kinds of things can you do to improve ? One approach could be to try and correct the imbalance. You can do it by specifying class weights – or you can do it via resampling – will talk about both.
  + If you have evaluated multiple models, yet another thing you can do is to try and create a voting classifier to combine the models and see if that improves the results.
  + Another approach could be to get more data. Can you use the notebook the Kaggle contributor used to grab data from other years ? (This data is 2015 I believe)
* Conclusion
  + What did you conclude ?
  + What other things can be done to improve quality that you did not get to do in the project ?

The data is a part of the 2015 Behavioral Risk Factor Surveillance System annual survey done by the CDC. The subset I am using was made by Alex Taboul on github, in which he consolidated the original dataset to include only the columns he determined to be relevant to diabetes detection. The original dataset contains responses from 441,455 individuals and has 330 features, while the dataset I used is comprised of 253,680 responses and 22 variables total. The features consist of things like high cholesterol or blood pressure, smoker status, age, BMI, sex and income. All of the features are categorical, ranging from binary to 30, except for BMI.

I am using two different versions of the dataset, one binary which contains diabetic (which is a combination of pre-diabetic and diabetic) & non-diabetic, and one multiclass which separates the three.

My goal for this project was to make two classifiers to predict diabetes, one binary and one multi-class.

EDA

The dataset has a pretty large imbalance, with over a 6:1 ratio of non-diabetic to diabetic. This can impact the results as the model will be much better trained on non diabetic compared to diabetic/pre-diabetic. This could lead to false negatives, in which someone who should be categorized as 1 (diabetic), may be categorized as 0.

The dataset I used was cleaned, so there were no nulls. There are duplicates in the dataset but I decided not to remove them, as with a medical dataset like this, each record is valuable in training the model. Also, duplicate records do not necessarily indicate error in this case as some individuals may have the same records.

I looked at the correlation scores of each feature with the label, and all of them seem to be correlated with the label, with the lowest correlation being with the feature that indicates if the patient had healthcare or not.

The correlation scores were not definitive enough, so I decide to use the chi2 test with an alpha of 0.01 to determine the feature importance. This resulted in all features except one being significant.

Modelling

The plan for this project was to build baseline classifiers for both the binary and multiclass datasets. I decided on using RandomForest and HistGradientBoost, which is much faster than [GradientBoostingClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html) for big datasets (n\_samples >= 10 000).

I split the data using a 20% test size and stratified with label column. I then did feature selection using chi2 and an alpha value of 0.01 to transform the training data, which resulted in one dropped feature, any healthcare.

I then used gridsearchcv to tune the hyperparameters for each model. For the random forest models, I tuned the number of estimators, max depth of each tree, and max features (which determines what portion of the total features are used to randomly train each tree) . For the HistGradientBoost, I tuned the max itererations, max depth, minimum samples per leaf, and learning rate parameters. I used a 5 fold cross validation and the f1 weighted metric. I chose this metric because accuracy is not very reliable for imbalanced data sets and precision + recall are better measures. The F1 score is a harmonic mean of the precision and recall, with its best value at 1 and worst score at 0. I chose the weighted version to provide the correct weights to the imbalanced classes.

Discussion

I started with the hypothesis that I can build a diabetes classifier for both the binary and multiclass diabetes datasets. The results of this project show that it is possible, but I definitely think that there is room for improvement. Correcting the data imbalance would be the first thing I would do.

Both the algorithms I used for this project allow me to specify custom weights for the labels to account for imbalance, so changing those would be the first step in helping imbalance. I have also read of something called synthetic resampling (SMOTE), which I have read also helps with correcting imbalance.

Then I would investigate acquiring additional data, whether it be more samples or more features. I would also look at the features in the original dataset and see if any were left out that could potentially better the model.