/\* prova random forest \*/

proc hpforest data= scmm.imprese;

target classe\_di\_valutazione /level=nominal;

input regione forma\_giuridica/level=nominal;

input ricavi numero\_dipendenti indice\_di\_liquidita capitale risultato\_di\_esercizio/ level=interval;

ods output FitStatistics=fitstats(rename=(Ntrees=Trees));

run;

data fitstats;

 set fitstats;

 label Trees = 'Number of Trees';

 label MiscAll = 'Full Data';

 label Miscoob = 'OOB';

run;

proc sgplot data=fitstats;

 title "OOB vs Training";

 series x=Trees y=MiscAll;

 series x=Trees y=MiscOob/lineattrs=(pattern=shortdash thickness=2);

 yaxis label='Misclassification Rate';

run;

Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.

This approach utilizes the usage of bootstrapping in the random forest. Since the bootstrapping samples the data with the possibility of selecting one sample multiple times, it is very likely that we won’t select all the samples from the original data set. Therefore, one smart decision would be to exploit somehow these unselected samples, called out-of-bag samples.

Correspondingly, the error achieved on these samples is called out-of-bag error. What we can do is to use out-of-bag samples for each decision tree to measure its performance. This strategy provides reliable results in comparison to other validation techniques such as train-test split or cross-validation.

 **Out-of-bag Error in Random Forests**

Generally, in machine learning and data science, it is crucial to create a trustful system that will work well with the new, unseen data. Overall, there are a lot of different approaches and methods to achieve this generalization. Out-of-bag error is one of these methods for validating the machine learning model.

OOB Out of Bag is a way of validate random forest The **out-of-bag** (OOB) error is the average error for each calculated using predictions from the trees that do not contain in their respective bootstrap sample. **Random forest utilizes another compelling technique called bootstrapping**. This is a statistical method that we can use to reduce the variance of machine learning algorithms. In short, it takes the original data set and creates a subset for each decision tree. Also, this method samples subsets randomly with the possibility of replacement. It means that we can select one sample or value multiple times.

Finally, to generalize the procedure, random forest limits the number of variables used for constructing decision trees. When we want to split the node of a decision tree, it considers only a subset of randomly selected variables (features). In the popular machine learning library called sci-kit-learn, this hyperparameter of random forest is known as “max\_features”. Usually, the best practice is to set it to

* for classification
* for regression

where  is the total number of variables, and  is the number of variables selected randomly for a particular node split.

[**https://www.developersmaggioli.it/blog/classificatori-non-lineari-alberi-decisionali-e-foreste-casuali/**](https://www.developersmaggioli.it/blog/classificatori-non-lineari-alberi-decisionali-e-foreste-casuali/)

**Foreste casuali**

Nel mondo reale che cos’è una foresta? Un insieme di alberi.
Nell’ambito dell’intelligenza artificiale che cos’è una **foresta casuale** (o **random forest**)? Un modello di classificazione che sfrutta numerosi alberi decisionali per ottenere un risultato complessivo migliore (migliore generalizzazione, minore overfitting) rispetto a quello che si otterrebbe con ognuno degli alberi presi singolarmente. In generale la metodologia di utilizzare un insieme di modelli che concorrono ad ottenere un risultato migliore è detta **apprendimento ensemble**, di cui le foreste casuali sono uno dei casi più famosi.

Come si crea una foresta casuale? Innanzitutto si decide quale dovrà essere il numero di alberi che compone la nostra foresta. Più alto è il numero di alberi, più il carico computazionale aumenterà, quindi è necessario arrivare (come sempre) ad un compromesso tra accuracy e sforzo di elaborazione. Successivamente selezioniamo casualmente un insieme di esempi dal nostro dataset. Questo sottoinsieme lo utilizziamo come training set per un albero decisionale. Si ripete la scelta casuale del sottoinsieme di esempi e la generazione di un nuovo albero, fino a quando non raggiungiamo il numero di alberi che compongono la nostra foresta casuale.

Una volta ottenuta la foresta casuale, come eseguiamo una predizione? Si eseguono le predizioni di ogni singolo albero decisionale che compone la foresta, ed infine si utilizza una regola per decidere la classificazione finale in base ai risultati ottenuti da tutti gli alberi. Da questo si capisce l’importanza del numero di alberi che compone la foresta: maggiore è il numero di alberi, migliore sarà il modello, ma maggiore sarà anche lo sforzo di calcolo.