Notes

# Basic Model

* **Common assumptions** 
  + **Open Date** used to estimate number of year from open date and to 2014 (guessed target year)
  + **City, City Group, Type** treated as **categorical variables** encoded but treated as numeric and not as factors
  + **P1-P37** treated as numeric variables
* **6-fold + bootstrap resampling**
* **Applied caret’s *nearZeroVar* function** 
  + Performed as **1,855,**273.26 on **leaderboard**
  + **Winner** model is **Enet\_Reg** (cross validation RMSE = **2,361**,006)
  + **Second** model is **RandomForest\_Reg** (cross validation RMSE = **2,362**,509)
  + **Third** model is **SVM\_Reg** (cross validation RMSE = **2,370**,142)
  + **Fourth** model is **PLS\_Reg** (cross validation RMSE = **2,371**,511)
  + **Fifth model** is **RobustLinearReg** (cross validation RMSE = **2,395**,484)
* **Removed only zero variance predictors[[1]](#footnote-1)**
  + Performed as **1,798,733.**58607on **leaderboard** (**improving 56,539.67394** the best performance of the winner model applying caret’s nearZeroVar function)
  + **Winner** model is **BaggedTree\_Reg** (cross validation RMSE = **2,361,869**)
  + **Second** model is **RandomForest\_Reg** (cross validation RMSE = **2,364,129**)
  + **Third** model is **Average** (cross validation RMSE = **2,450,247**)

Notice that **BaggedTree\_Reg** and **RandomForest\_Reg** don’t require feature scaling. Moreover it should be interesting **leaving all predictors** **un-scaled** except zero variance ones:

* **Winner** model is **BaggedTree\_Reg** (cross validation RMSE = **2,321,782**); performed as **1,907,954** on **leaderboard**
  + **Learnt that for BaggedTree\_Reg** **feature scaling is important**
* **re-do w/** removePredictorsMakingIllConditionedSquareMatrix = F, removeHighCorrelatedPredictors = F, featureScaling = T; bagged tree submission (mySub\_bagged\_tree.csv) performed on leaderboard as **1,865,466.33115; that should mean that using removePredictorsMakingIllConditionedSquareMatrix = T, removeHighCorrelatedPredictors = T is good thing ever!**
* **Second** model is **RandomForest\_Reg** (cross validation RMSE = **2,322,125**); performed as **1,789,447.28778** on **leaderboard**

# Basic Model improvement #1: boosting like & boosting

We can try to create a new feature as the prediction of the winner model both on the test set and the training set (6-folds-like procedure) and using the second model to predict on test set

* First “**tip**” feature has been build on **Enet\_Reg** applying caret’s *nearZeroVar* function both in test set and training set (6-folds-like procedure) while the predicting model is a **BaggedTree\_Reg** removing only zero variance predictors ( removePredictorsMakingIllConditionedSquareMatrix = F, removeHighCorrelatedPredictors = F, featureScaling = F)
* performed as **1,839**,573 on **leaderboard**
* **re-do w/** removePredictorsMakingIllConditionedSquareMatrix = T, removeHighCorrelatedPredictors = T, featureScaling = T; (mySub\_tip\_1.csv) performed on leaderboard as **1,865,326.24078**

I fatti dimostrano che, se non ci sono bugs, e’ piu’ efficace la tecnica di boosting invece che quella di tipping (**1,839**,573 vs **1,795,587 : +2.5%)**

**The traditional Gradient Boosting for Regression**

1. Select tree depth, D, and number of iterations, K
2. Compute the average response, y, and use this as the initial predicted value for each sample
3. for k = 1 to K do
4. **Compute the** **residual**, the difference between the observed value and the current predicted value, for each sample
5. **Fit a regression tree of depth, D, using the residuals as the response**
6. Predict each sample using the regression tree fit in the previous step
7. Update the predicted value of each sample by adding the previous iteration’s predicted value to the predicted value generated in the previous step
8. end

Following this approach

* first we compute the response both in training set (6-folds-like procedure) and test set w/ **BaggedTree\_Reg** removing only zero variance predictors (removeOnlyZeroVariacePredictors=T, removePredictorsMakingIllConditionedSquareMatrix = F, removeHighCorrelatedPredictors = F, featureScaling = F)
* we compute the residual, i.e. the difference between the observed value and such prediction in train set;
* we fit an **Enet\_Reg** applying caret’s *nearZeroVar* function using the residuals as response;
* we predict the test set using the **Enet\_Reg;**
* we update the predicted value adding to the previous prediction (**BaggedTree\_Reg**) to the predicted value of residuals
* performed as **1,858**,328 on **leaderboard**
* **re-do w/ removeOnlyZeroVariacePredictors=T,** removePredictorsMakingIllConditionedSquareMatrix = T, removeHighCorrelatedPredictors = T, featureScaling = T; (mySub\_boost\_1.csv) performed on leaderboard as **1,755,088.64112**
* **re=do w/ RandomForest\_Reg instead of BaggedTree\_Reg:** performed on leaderboard as **1,788,992.76965**
* **re-do fitting also residuals residuals (y:RandomForest\_Reg,res: Enet\_Reg, res2:** **BaggedTree\_Reg):** performed as **1,795,587**.03120 on leaderboard
* **re-do fitting also residuals residuals** 
  + **y: BaggedTree\_Reg,res: Enet\_Reg, res2:** **RandomForest\_Reg (applying caret nearZeroVar function)**
  + performed as **1,861,892.03046**on leaderboard

I fatti dimostrano che, se non ci sono bugs, il primo modello da usare e’ **BaggedTree\_Reg poi** I residui vanno futtati con **Enet\_Reg.**

**Inoltre, modelli a 3 livelli (BaggedTree\_Reg,res / Enet\_Re / RandomForest\_Reg ) sembrano meno performanti di quelli a 2 livelli**

# Basic Model improvement #2: guessing categorical variables among P1-P37

We can try to guess one at the time one of P1-P37 as categorical, encoding it, and looking at the performance (both cross validation and test set)

* grid 37 rows
* each row: who’s categorical among P1-P37, winner model, best.perf xval
* **applying caret’s *nearZeroVar* function and other defaults**
* **on the Pi w/ highest probability re-do w/ removeOnlyZeroVariacePredictors=T**

# Bugs

Attenzione alla riga

pred = ifelse(pred >= 1150 , pred , 1150)

> min(y)

[1] 1149870

1. removePredictorsMakingIllConditionedSquareMatrix = T, removeHighCorrelatedPredictors = T, featureScaling = T [↑](#footnote-ref-1)