2013 American Community Survey

Thomas Leni, Carlo Sanzogni
mat. 925254, mat. 935799

April 9, 2021

Contents

List of Figures

List of Tables

Abstract. The aim of the project is to evaluate how the target variable, FINCP (family income of the past 12 months), is influenced from the heterogeneous group of the other variables. In order to study this, the Linear Model regression is run and then, in order to achieve better results, some extensions are investigated: the Ridge Regression, the LASSO Regression and the Elastic Net. A grid search on the different hyperarameters has been performed but no improvement in performances has been spotted.

1 Dataset Description and Pre-processing

The dataset analyzed is the *2013 American Community Survey*, which contains more than 100000 records, coming from American Community Survey, an on going survey from US Census Bureau.

For the project are considered only the housing data in which each row is a housing unit and describes the characteristics concerning the status of the houses and their inhabitants. Like all survey dataset, there are a multitude of variables, so it is necessary to make a selection of these; first of all, only one territorial variable is kept in order not to have redundancy information. After, the variables too correlated with that target are eliminated as HINCP (Household income). Finally, the adjustment factors and the replicated weights, *WGPT*, are eliminated because they are not useful for the analysis.

Due to the fact that the dataset is made up with different survey questionnaires jointed together, there are many missing values. To avoid the problem only the observations in which the target variable is present are kept, taking down the rows to approximately 90000; for the other variables the remaining null values are only a few and so replaced by zero.

Despite the large selection made, is still present a too high number of explanatory variables, more than 150, so it became necessary to perform a PCA. Thanks to this technique, as will be explained in the next section, is possible to reduce the number of variables without losing relevant information.

Only one last operation has been performed on data before starting applying the actual models: data need to be scaled. This operation has been performed in four steps: first an absolute scaler is trained on the regressors of the training set, it means that takes the minimum and maximum for each column; secondly with these values either the variables of training and test set are scaled; then both steps are repeated for the target variable again taking the limits values only from the train in order to scale all observations with the same range.

2 Theoretical material

2.1 PCA - Principal Component Analysis

As mentioned before the reduction of dimensionality is an important technique for pre-processing of supervised and unsupervised learning; it is used in order to eliminate redundant and irrelevant information in the dataset, preserving however the most important disclosure. This is important because having data with lower dimensional helps to train an algorithm without wasting time.

Therefore the Principal Component Analysis is part of the linear transformation problems and it is applied for the extraction of the characteristics of a multivariate set, in order to obtain a new subspace with dimension equal or smaller than the starting one.
The PCA is composed by five steps:

- Standardization: the aim is to standardize the range of the continuous initial variables so that each one of them contributes equally to the analysis. It is really important because if there are large differences between the ranges of initial variables, those variables with large ranges will dominate over those with small ranges. Hence, transforming the data to comparable scales can prevent this problem.
  From a mathematical point of view the equation is:

  $$z = \frac{value - mean}{std.deviation} \tag{1}$$

  In this way all the variables will be transformed to the same scale

- Covariance matrix computation: the aim is to understand how the variables of the input dataset are varying from the mean with respect to each other, so if there is a relationship among them. This allows to understand if variables are highly correlated in such a way that they contain redundant information.
  The covariance matrix is a p Ã p symmetric matrix, where p is the number of dimensions, that has as entries the covariances associated with all possible pairs of the initial variables. If the covariances as entries of the matrix tell about the correlations between the variables, there are two possible scenarios: if it is positive then the two variables are correlated, if it is negative instead then the two variables are inversly correlated.

- Compute eigenvectors and eigenvalues of the covariance matrix to identify the principal components: eigenvectors and eigenvalues are the linear algebra concepts that allow to compute from the covariance matrix in order to determine the principal components of the data.
  Principal components are new variables that are constructed as linear combinations or mixtures of the initial variables. These combinations are done in such a way that the new variables are uncorrelated and most of the information within the initial variables is squeezed or compressed into the first components. This process allows to reduce dimensionality without losing much information.
  As there are as many principal components as there are variables in the data, principal components are constructed in such a manner that the first principal component accounts for the largest possible variance in the data set. The second principal component is calculated in the same way, with the condition that it is uncorrelated with the first principal component and that it accounts for the next highest variance.

- Feature vector: computing the eigenvectors and ordering them by their eigenvalues in descending order, allow to find the principal components in order of significance. Now there is the choose whether to keep all these components or discard those of lesser significance, and form with the remaining ones a matrix of vectors that we call Feature vector.
  So, the feature vector is simply a matrix that has as columns the eigenvectors of the components that are decided to keep. This makes it the first step towards dimensionality reduction with only p dimensions.

- Recast the data along the principal components axes: the aim is to use the feature vector formed using the eigenvectors of the covariance matrix, to reorient the data from the original axes to the ones represented by the principal components. This can be done by multiplying the transpose of the original data set by the transpose of the feature vector.

$$FinalDataSet = FeatureVector^T \cdot StandardizedOriginalDataSet^T \tag{2}$$

2.2 The Linear Regression

The Linear Regression is one of the simplest methods for explain the relationships between some independent variables and describe their effects on the target variable of the problem.

In general it is based on the Ordinary Least Squares which is a statistics method for estimating the unknown parameters of the regressors. This procedure is based on the minimization of the square of the sum of the residual between the regression line and data points. From the residuals is also possible to derive the variance, or Mean Square Error, of the model:

$$s^2 = \frac{\sum e_i^2}{n - p - 1} \tag{3}$$

where e are the residuals presented before $(y_i - \hat{y}_i)$, n is the number of observations and p the number of explanatory variables. Important is to remember that from this could also be computed the so called standard error simply by taking the square root.

Finally is important that $\hat{y}_i$ is computed solving the equation:

$$\hat{y}_i = b_0 + b_1 x_i 1 + b_2 x_i 2 + ... + b_p x_i p \tag{4}$$

where $i = 1...n$, and the b are the estimation of the real coefficient β.

2.3 Regularization methods: Ridge regression, Lasso regression and the Elastic Net

Ridge and Lasso regression are some of the simple techniques to reduce model complexity and to fix over-fitting and multi-collinearity which may result from simple linear regression. Here there is an explanation of the two regressions:

- Ridge regression: in ridge regression, the cost function is altered by adding a penalty equivalent to square of the magnitude of the coefficients.

  $$\sum_{i=1}^{M}(y_i - \hat{y_i})^2 = \sum_{i=1}^{M}(y_i - \sum_{j=0}^{p} w_j \times x_{ij})^2 + \lambda \sum_{j=0}^{p} w_j^2 \tag{5}$$

  Hence it puts constraint on the coefficients w; the penalty term lambda regularizes the coefficients such that if the coefficients take large values the optimization function is penalized. So, ridge regression shrinks the coefficients and it helps to reduce the model complexity and multi-collinearity.

- Lasso regression: the cost function for Lasso regression is the following:

  $$\sum_{i=1}^{M}(y_i - \hat{y_i})^2 = \sum_{i=1}^{M}(y_i - \sum_{j=0}^{p} w_j \times x_{ij})^2 + \lambda \sum_{j=0}^{p} |w_j| \tag{6}$$

The different with the previous regression is that instead of taking the square of the coefficients, magnitudes are taken into account. This type of regularization can lead to zero coefficients so some of the features are completely neglected for the evaluation of output. Lasso regression not only helps in reducing over-fitting but it can help in feature selection. Like Ridge regression the regularization parameter lambda can be controlled and feature selection using Lasso regression can be depicted well by changing the regularization parameter.

- Elastic Net: this last technique is a synthesizer of the two previous methods, in fact its cost function is the following:

$$\widehat{\beta_{net}} = argmin_b \frac{\sum_{i=1}^{n} (y_i - x_i b)^2}{2n} + \lambda \left(\frac{1-\alpha}{2} \sum_{k=1}^{k} b_k^2 + \sum_{k=1}^{k} |b_k| \right) \tag{7}$$

  It is based on two parameters, the already cited Lambda and a new one Alpha, which lays between 0 and 1 and, as could be understand by the function above, from this value is possible to understand if the model is closer to Ridge or LASSO.

Both methods allow to use correlated predictors, but they solve multicollinearity issue differently: in ridge regression, the coefficients of correlated predictors are similar; instead in lasso, one of the correlated predictors has a larger coefficient, while the rest are zeroed. Lasso tends to do well if there are a small number of significant parameters and the others are close to zero; on the contrary ridge works well if there are many large parameters of about the same value. As explained in the last part the Elastic Net is a solution when there is no certainty of which of the methods could better solve the problem.

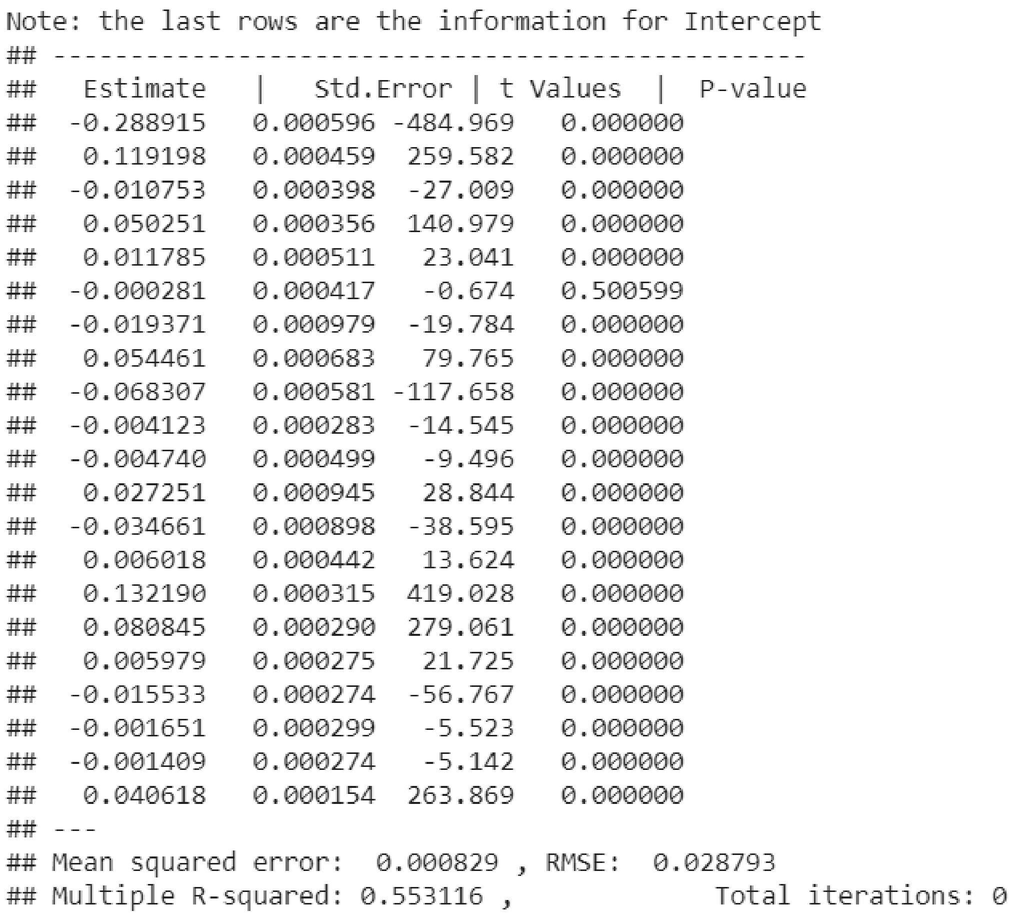
3 Principal results and comments

In this section there is an in depth analysis of the dataset with different supervised techniques. The principal goal is to find a model which can perform good results on the objective variable and at the same time is scalable with growing data. In the next section there is the analysis of each method with the consequent choice of the most suitable for the dataset used.

3.1 Linear regression

As presented in literature section first is trained a simple Multivariate Linear Regression. Despite its simplicity with this type of data, also due to the huge preprocessing (PCA and normalization), the results are not so bad: as could be seen in [Figure 1] almost all the principal components considered are statistically significant; both Mean Squared Error and RMSE on the training set achieve a low level, but this will be more clear when compared with the following methods; lastly the R squared reaches a 55,3%, which although not being very high, is not that bad.

Important is to compare the results of training set with the ones of the test. The two are obtained by dividing the original dataset with the proportions of 70/30. These are quite satisfying; as shown in [Table 1] no remarkable differences have been spotted, on the contrary the R2 of the train and test set are very similar and this means that the model is no overfitted.



```
Note: the last rows are the information for Intercept
## ------------------------------------------------
##   Estimate   |   Std.Error | t Values  |  P-value
##  -0.288915   0.000596 -484.969   0.000000
##   0.119198   0.000459  259.582   0.000000
##  -0.010753   0.000398  -27.009   0.000000
##   0.050251   0.000356  140.979   0.000000
##   0.011785   0.000511   23.041   0.000000
##  -0.000281   0.000417   -0.674   0.500599
##  -0.019371   0.000979  -19.784   0.000000
##   0.054461   0.000683   79.765   0.000000
##  -0.068307   0.000581 -117.658   0.000000
##  -0.004123   0.000283  -14.545   0.000000
##  -0.004740   0.000499   -9.496   0.000000
##   0.027251   0.000945   28.844   0.000000
##  -0.034661   0.000898  -38.595   0.000000
##   0.006018   0.000442   13.624   0.000000
##   0.132190   0.000315  419.028   0.000000
##   0.080845   0.000290  279.061   0.000000
##   0.005979   0.000275   21.725   0.000000
##  -0.015533   0.000274  -56.767   0.000000
##  -0.001651   0.000299   -5.523   0.000000
##  -0.001409   0.000274   -5.142   0.000000
##   0.040618   0.000154  263.869   0.000000
## ---
## Mean squared error:  0.000829 , RMSE:  0.028793
## Multiple R-squared: 0.553116 ,            Total iterations: 0
```

Figure 1: Summary of Linear Model

3.2 Regularization and extensions of the model

In order to improve the performance of the model further extensions are developed: some hyperparameters are added and validated but unluckily as will be clear from the following results this attempt does not lead to significant improvement.

In depth three types of parameters are investigated:

- the Intercept: due to the fact that all variables have been normalized was conjectured to not having an intercept parameter and so a centred model was possible; unluckily the best model spotted disproved this hypothesis;
- the Regularization Parameter Lambda: as seen in the theoretical section this parameter weights the penalties, in the search five values are studied: 0.0001, 0.001, 0.01, 0.05, 0.1;
- the Parameter Alpha: this parameter is train with five values between 0, Ridge Regression and 1, LASSO Regression.

	Training Set	Test Set
MSE	0.0008	0.0009
RMSE	0.0288	0.0303
R SQUARED	0.5531	0.5495

Table 1: Training/Test set comparison for Linear Model

A Grid Search has been performed validating all possible combinations of parameters, so 50 models are compared, taking as goal the minimum possible RMSE; surprisingly an unexpected result appears: the Intercept is present, the Regularization Parameter Lambda is the minimum between the selected (0.0001) and the Parameter Alpha is 0. In other words the resulting model is a Ridge Regression with a Lambda Parameter almost equal to zero, so the same Linear Regression already discussed in the previous subsection.

Due to the unexpected results another model has been trained, an Elastic Net with Lambda= 0.01 and Alpha=0.2 but, to confirm the previous results, as could be observed in [Table 2] all performance evaluators either on training set and on test get worst.

	Training Set	Test Set
MSE	0.0009	0.0009
RMSE	0.0303	0.0305
R SQUARED	0.5017	0.4987

Table 2: Training/Test set comparison for Elastic Net

Analyzing in depth the values suddenly could be noticed that under the Error point of view there are no significant decreasing in performance, the Errors of Linear Model and this Elastic Net are very similar but switching on the R Squared there is an important fall, from 0.55 to 0.50. This confirm what found during the parameters optimization: adding bias to the Linear Model, no matter if with Ridge, LASSO or Elastic Net approach, does not lead to better performances.

4 Conclusions

In conclusion could be state that the Linear Model is the best between the taken into account. Even if the R Squared never reaches very high levels, as already explained, adding hyperparameters optimized by searches does not improve the output.

However the goal of this project was not only composed by the model but also by its scalability at growing of data size. This objective is accomplished using PySpark technology, a Spark implementation for Python, which permits to deal with distributed systems and so to make different machines operates simultaneously. This pipeline is all constructed using PySpark objects so, even not dealing with a multiple machines environment in this project, could simply be adapted without many changes in the case of a bigger data size.

5 References

References

[1] https://www.kaggle.com/census/2013-american-community-survey

[2] https://www.datacamp.com/community/tutorials/tutorial-ridge-lasso-elastic-net

[3] https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b

[4] https://lorenzogovoni.com/pca/

[5] https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[6] https://towardsdatascience.com/a-brief-introduction-to-pyspark-ff4284701873

[7] https://www.lorenzogovoni.com/tre-tecniche-di-regolarizzazione-ridge-lasso-ed-elastic-net/

[8] https://runawayhorse001.github.io/LearningApacheSpark/regression.html

[9] https://towardsdatascience.com/distributed-deep-learning-pipelines-with-pyspark-and-keras-a3a1c22b9239

[10] https://projector-video-pdf-converter.datacamp.com/14989/chapter4.pdf