Kaggle Bike Project Report

Michelle Evans Bouchet, Nayef Ghattas, Yiren Wang

1 Pre-processing

First we loaded the data. In the original data set, we have 13 features and 10886 samples. However, the date feature seems to be harmful; indeed, the training set contains samples with dates from the 1st to the 20th and the test set contains the dates from the 21st to the 30th. Therefore, the remaining features are the season, the year, the month, the hour of the day, the temperature, the apparent temperature, humidity, holiday, weekday, working day, weather situation, and wind speed.

In order to better appreciate the features we plotted the data in histograms, comparing the raw data to the scaled and normalized data (c.f. Appendix). We also checked for the minimum and maximum of bikes borrowed; we'd like our predictions to stay within this range. We're scaling the data, but not normalizing the sample vectors as it is harmful towards binary format data. Furthermore, certain features such as the hour, month, and season are cyclic. For example, December and January are closer in reality than in the case of regression. In order to implement this, we have to replace the feature with two of them, one k.cos(feature) and k.sin(feature), k being a coefficient to scale the data. We tried PCA to select the more important features to the 14 features we have in total. The number of feature (13) being a lot smaller than the number of samples (10886), we might not have a need to implement feature selection or extraction. This is confirmed with the scree graph (cf Appendix) where we cannot visually see an elbow that will allow us to ignore certain features.

2 Performance

We used 10-layer cross-validation to evaluate learning algorithms. We defined a rmsle function to measure performance during cross-validation. We also made a scorer associated to it so as to use it a a scoring for GridSearchCV when trying to fine tune the parameters.

For all the algorithms, we list both the errors at cross-validation and validation. All errors are calculated with RMSLE.
Performances obtained are as follows :

2.1 Linear Regression

Cross-validation : 1.30242850965 **Validation** : 1.35052

The error is rather consistent between cross-validation and validation. The latter is slightly higher because the algorithm might be slightly over-fitting. In order to reduce over-fitting, we can use regularized regression :

2.2 Regularized Regression

The purpose of regularized regression is adding an penalty to the loss function through α (the coefficient for the vector of the learned parameters). It is interesting then to find α through GridSearchCV for each of the following types of regularized regression.

2.2.1 Lasso

This method supposedly allows us to eliminate sparse coefficients to ensure that we are not learning the noise in the data. Nonetheless, the discrepancy between our results shows that we are over-fitting.
Cross validation : 1.24224158936. **Validation** : 1.32830

2.2.2 Ridge

Ridge Regression obtained a worse score than Lasso which is understandable as it ridge regression cannot reduce to zero the unimportant coefficients; thus it has a tendency to over-fit.
Cross validation : 1.30275065064 **Validation** : 1.34903

2.2.3 Elastic Net

Logically speaking, certain features such as temperature and apparent temperature are highly correlated which implies that elastic net might be a better choice than lasso. This is to keep groups of correlated predictors in the model.We obtain better results than we the other regularization methods.

Cross validation : 1.28109751646 **Validation** : 1.31219

2.3 Nearest Neighbor Regression

We tuned parameters with grid search CV, we obtained the best results with $n\_neighbours = 2$ and a $leaf\_size = 30$. The results obtained with scaling are worse; perhaps it is because it heightens the similarity between samples artificially. By raising the $n\_neighbors$ parameter we might improve performance by over-fitting less. Without scaling we obtain the following RMSL Error :
Cross validation : 0.412209721835 **Validation** : 0.53072

2.4 Random Forest Regression

Random forests led us to good results during cross-validation even when using default parameters (0.33). The optimal parameters are : $min\_samples\_leaf = 3$ and $n\_estimators = 16$.
Cross validation : 0.33 **Validation** : 0.47847

2.5 Support vector regression

We began by setting gamma and C by iterating with GridSearchCV over a large range of values and slowly reducing the step. The errors obtained between cross-validation and validation are consistent, so we are most likely not overfitting.
Cross validation : 0.77757 **Validation** : 0.77755

3 Final Model

For our final model we chose an algorithm we have not studied in class. Indeed, encouraged by the positive results we obtained (and a reasonable computing time, contrary to svm) we decided to look for other Tree based algorithm. The best results we obtained were with ExtraTreesRegressor.
Cross validation : 0.337957970738 **Validation** : 0.46332
We are probably overfitting to our training data; to counter it, we tried increasing the $min\_samples\_leaf$, the minimum number of samples that there should be at each leaf; this has our trees make less divisions and thus overfit less, but the results obtained at validation were slightly worse; however it remains the best model we've found.

4 Appendix

Figure 1: We can see a Gaussian-like distribution

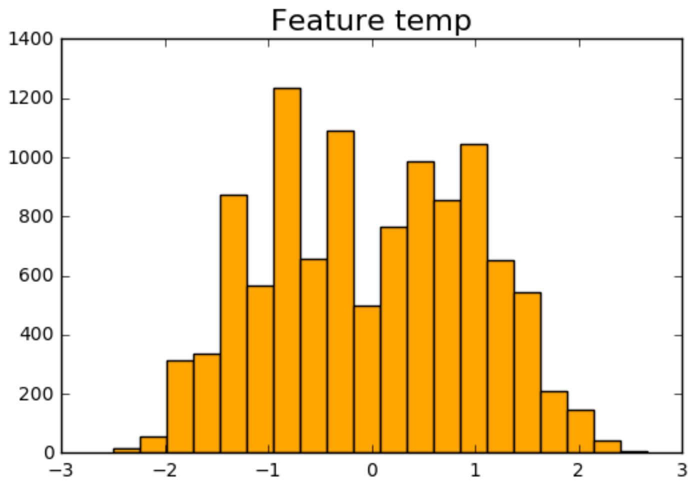


Figure 2: Number of bikes borrowed according to the temperature

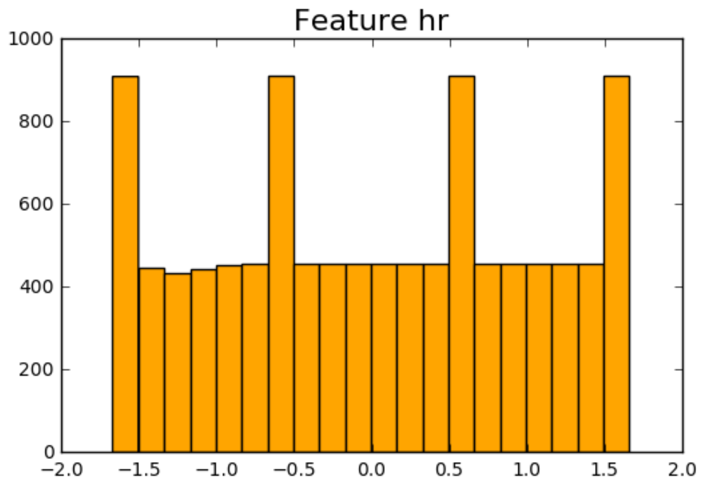


Figure 3: Number of bikes borrowed according to the humidity

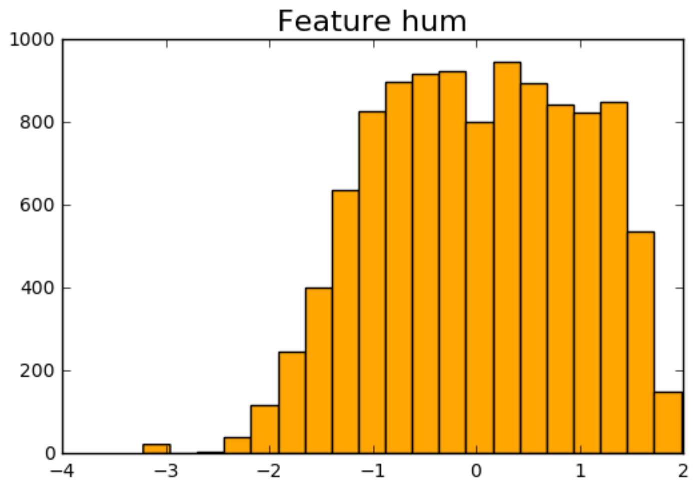
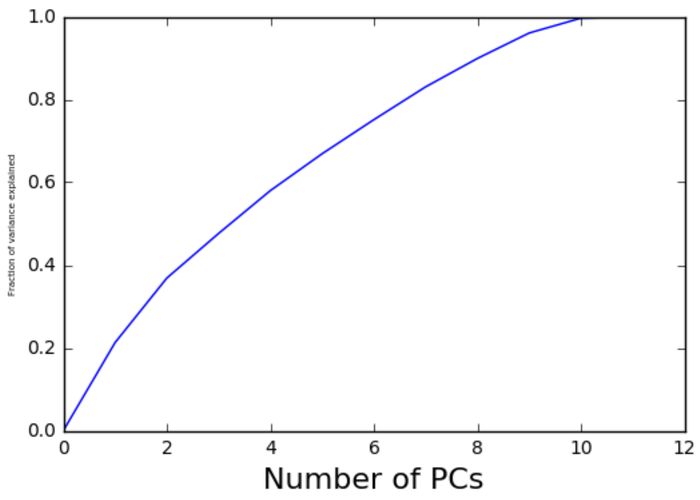


Figure 4: Scree curve - no marked elbows


Fraction of variance explained
Number of PCs