VAN ECCELPOEL
Théo

Advanced Machine Learning
Lab 1 – Linear Regression
Report & Analysis

In this report, I will analyze the lab I did on linear regression. Linear regression allows to predict the outcome of numerical variables.
Simple linear regression is based on assumptions (LINE), but we can relax some of these assumptions using several techniques such as regularization, polynomial features, robustness integration...
In this lab, we will see how linear regression can be tweaked to perform well on multiple datasets, which make this model very attractive, especially combined to its easily interpretable results. We will start from the basic model and evolve until best regression models for more complex datasets.

1) Simple Linear regression

Basic linear regression is the simplest form of linear regression, which has the advantage of being super easy to interpret, very transparent while still performing well on some datasets. However, this model is mainly restricted by its linear relationship, that makes it irrelevant for numerous cases.

2) Basis Function Regression

To relax this drawback, we can use basis functions, that transform our data into a higher-dimension space, where we can handle non-linear relationships such as logarithms, exponentials, polynomials etc.
In this lab, we focus on the polynomial transformation. In fact, it is still a linear regression as features are related through a linear relationship, but our features can handle non-linear data, in our cases, polynomial data.
This transformation is very interesting, as, depending on the degree of polynomial we use, we can come up with completely different models that may fit way more models!
However, as the complexity of the model increases, the overfitting increases, reducing our performance on the test set.

I learnt that there is a tradeoff to make, to balance model complexity, interpretability and performance on unseen data. To solve this issue, we could try cross-validation to select best complexity for unseen data or use a regularization parameter to penalize the complexity of the model.

3) Regularization

Linear regression is even more powerful when we implement regularization terms, such as the l0, l1 or l2 norms, that are more or less strict on model complexity.
In this lab, we focused on l1 and l2 norms, with Lasso and Ridge regularization.
The Ridge regularization penalizes the sum of squares of the model coefficients, whereas the Lasso regularization penalizes the sum of absolute value of the model coefficients.
It is interesting to notice that L1 norm is way stricter on its penalization, in the sense that it will drag the coefficients of non-useful features to zero, whereas the L2 norm is less strict, and will only reduce the coefficient of those features.

This lab was interesting as I had to implement the regularization "from scratch" in my python classes, and I noticed that the L1 norm was more difficult to implement than the L2 norm, as I had to use several loops and gradient descent to penalize the coefficients. This has impact on computation power, so when choosing between L1 and L2 norms, I may as well have a look at the computing resources I have available.

To use the best potential of regularization, I also learnt that the regularization parameter had a big role, and that I should test for several values of alpha to get to the best model for the data. This coefficient controls the strength of the penalty term and is therefore primordial to optimize.

With regularization, we solved the issue of features overfitting. However, linear regression can be very sensitive to outliers, that drag the features coefficients toward this outlier, decreasing the outcome prediction power.

4) Robustness

Linear regression is really powerful, as we can complexify a bit more the model to make it robust to outliers.
To do so, I learnt to implement a "weight" function, that will iteratively add a weight to each residual, while assigning lower weights to higher residuals.
I think this is very smart, as outliers will produce high residuals, but with the weight function (in our case, Huber or Bisquare functions), those residuals will count less in the estimation of our feature coefficient.
However, I believe that those weight functions should be applied only once we already have a decent model, as if we use this technique directly, we may not penalize enough bad models that predict badly.

5) Example

Finally, we try our models on a dataset, whose data look like a sinusoid curve.
It is interesting to notice that, despite trying regularized models and robust models, the performance of the linear regression does not really improve for this dataset. Sometimes, the simplest model is the best !
However, globally, we still observe a good prediction of our linear regression models, even on non-linear dataset…It perfectly got the shape of the data (seasonality), but struggles to capture some peaks. This shows that we may miss key features in our model to better fit to this dataset, or just that some irregular shocks appeared and our model couldn’t capture them.