CS725- Introduction To Machine Learning

Programming Assignment

The Effectiveness of Linear Regression Models

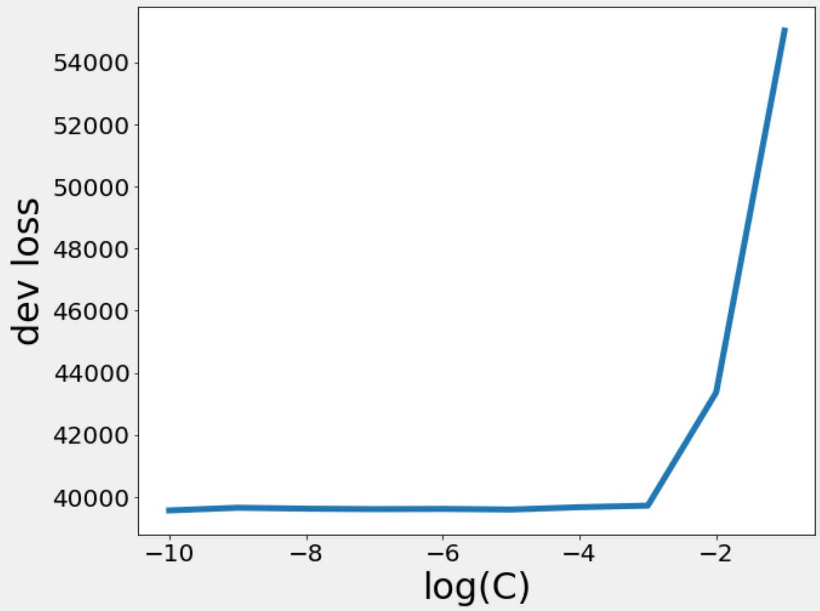
Sr No.	Name	Roll No.
1	Indradyumna Roy	214050004
2	Anish Mukundlal Chaurasiya	180260007 (Kaggle Team Name)
3	Sanket Mishra	194190004
4	Akshay Vilas Upasany	184190002
5	Manish Ashokrao Thombre	204100008

1. We obtain our best MSE losses by using a degree three polynomial basis function. We report the best MSE losses on the development set:
   a. With basis function :
      Analytical solution : train loss 2815.92 and dev loss: 3860.5
      Gradient descent solution: step 2004400 dev loss: 4737.657815662553 train loss: 3245.860815134
      Please note that the gradient descent was trained with lr=0.01, C=1e-8, batch_size=256, patience=100000. This trained for 2004400 runs for over 9 hours and had not yet reached convergence.

   b. Without basis function:
      analytical_solution train loss: 26021.21091519896, dev_loss: 39072.099491451976
      gradient_descent_soln train loss: 26326.625072571, dev_loss: 39313.92700887245
      In this case, gradient descent was trained with `lr=0.01, C=1e-8, batch_size=32, patience=1000`. Training reached convergence as per early stopping criteria. In this case we observe that the MSE loss, for both train and dev, using gradient descent is close to the analytical solution.

2. Gradient descent stopping criteria.
   a. We use early stopping for convergence during gradient descent. We use a patience parameter <2000> . After every iteration of gradient update, we compare the current validation loss with the minimum validation loss till that point. If the current validation loss is smaller than the minimum, then the best model parameters are updated. If the validation loss does not improve for no. of runs greater than patience value, then convergence is deemed to have been reached, and training is stopped.
   b. MSE losses on `dev.set` instances with early stopping
      step 30135 **dev loss: 39505.197368977046** train loss: 26798.475452298237 with max step: 100000 and patience parameter: 2000
   c. MSE losses on `dev.set` instances without the use of early stopping.
      step 100000 **dev loss: 39422.7837346846** train loss: 26756.492634122198
   d. In the above runs, we do not use a basis function in the interest of time. We note that while the run without early stopping achieves slightly lesser MSE loss on dev set, the run with early stopping achieves close to similar values in 30% of the training time.
   e. Early stopping code is implemented in the do_gradient_descent function.

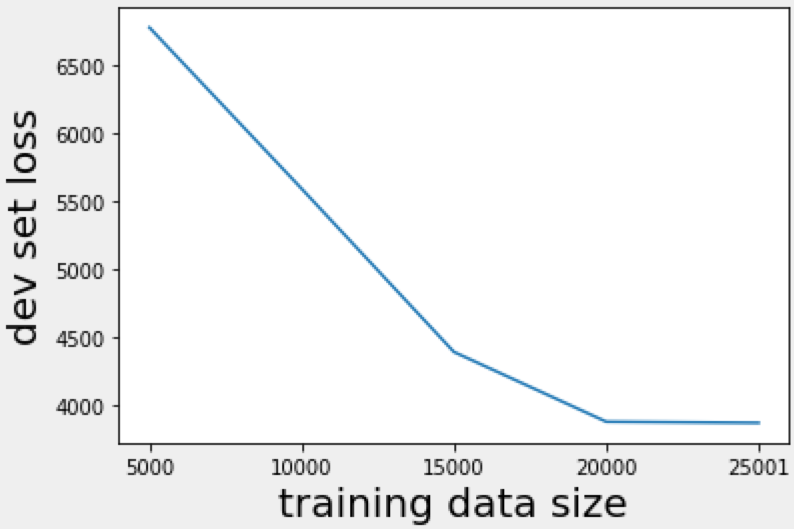
3. Effect of regularization:



   a. We plot the MSE on dev set for values of C:
      `C=[0, 0.1, 0.01, 0.001, 0.0001, 1e-05, 1e-06, 1e-07, 1e-08, 1e-09, 1e-10]`
      `C=0  dev_loss: 39605.060629685526` `train_loss=26933.874259047763`
   b. We obtain these results by training without any basis functions, using gradient descent.

4. Basis Functions:
   a. Polynomial Basis Function : We implement two variations of the polynomial basis function .
      i. First variation computes the second degree polynomial of all the features. More specifically, each input to the bsis function is a data point with 12 features, and the output of the function is a feature of length 144 - obtained by multiplying all nC2 pairs. We get an MSE of <`16717.980337151443` .> on the dev set using this.
      ii. Second variation computes the third degree polynomial of all the features. Here, given an input datapoint with 12 features, the basis function outputs a feature of length 1728 - obtained by multiplying all nC3 triples. We get an MSE of <`3860.567915319627` > on the dev set using this.
   b. Radial basis : We implement a radial basis kernel with 20 gaussian basis. Thus the 12 dimension features are mapped to 20 dimension features. We get an MSE of <67821.64390579217> on dev set using this.
   c. Radial basis is implemented in lines 87-95 and polynomial basis is implemented in lines 97-105.

5. Training Plots: We have added the implementation to the function `plot_trainsize_losses. Please note: these plots are obtained by using degree three polynomial basis function.`



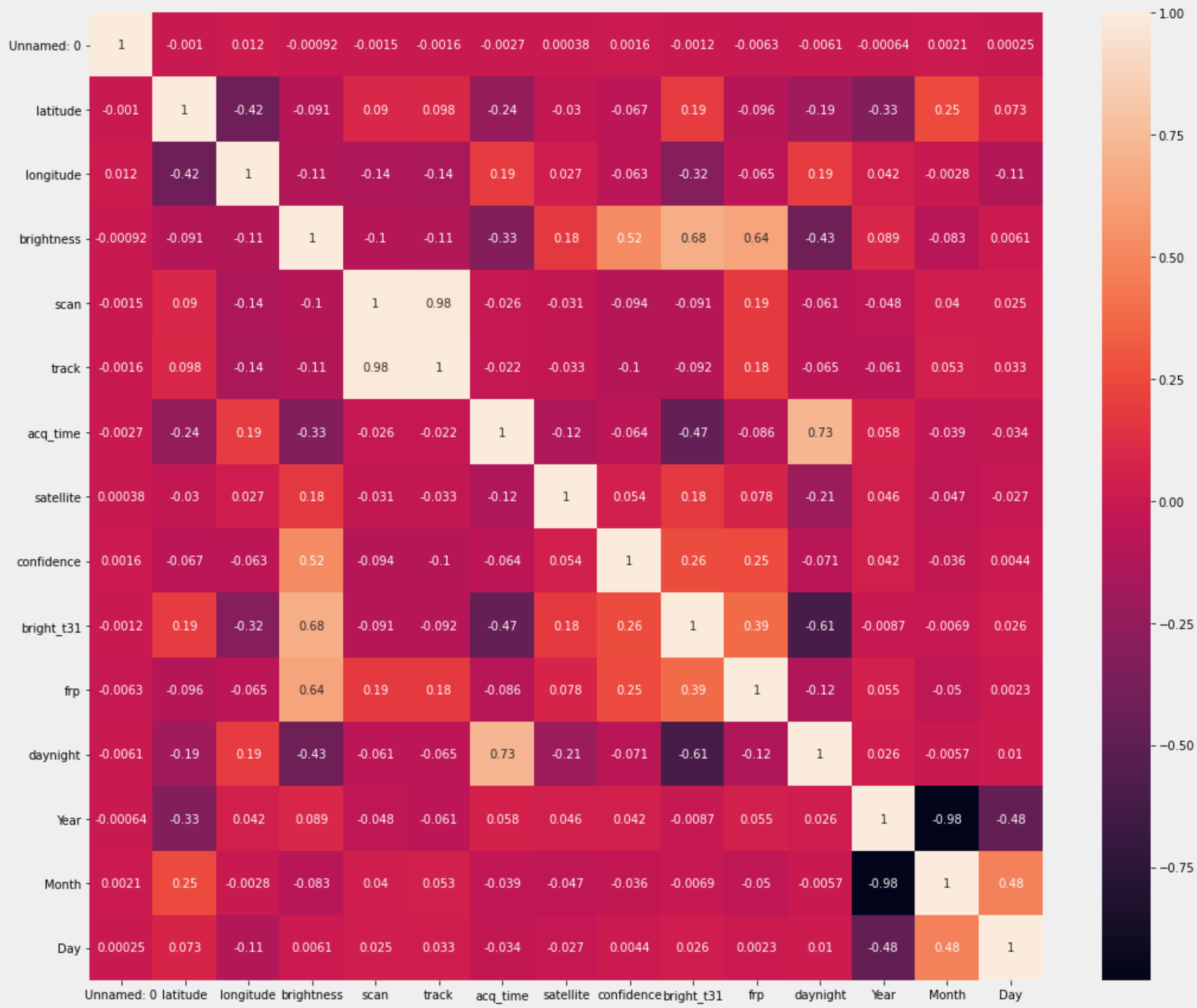
6. Feature Importance

Unnamed: 0	25001
latitude	13364
longitude	13585
brightness	1524
scan	39
track	11
acq_date	103
acq_time	171
satellite	2
instrument	1
confidence	101
version	1
bright_t31	739
frp	3782
daynight	2

dtype: int64

The above table lists the features and the corresponding no of unique values the feature takes. As we can see from the above description of data that same version and instrument were used for all the samples. As there is no variance in the version and instrument columns we can simply remove these two columns from the table. Also correlation coefficient of these two variables with frp is very less.

Another important point to note down is Unnamed: 0 and Day have very small correlation with frp, which is less than 0.01. We can remove this feature/column from our model.We observed that including the least significant feature in our model increased the train loss. so we were sure they were least significant. Brightness feature has the highest correlation with frp.



Conclusion: **Least important** features: instrument, version, Unnamed:0, Day
Most important features : `Brightness`

`Given features:(top 5 row)`

	Unnamed: 0	latitude	longitude	brightness	scan	track	acq_date	acq_time	satellite	instrument	confidence	version	bright_t31	frp	daynight
0	0	-25.117	149.245	363.1	1.2	1.1	2019-12-08	0	Terra	MODIS	100	6.0NRT	316.6	102.6	D
1	1	-32.263	123.294	349.3	3.4	1.7	2020-01-03	500	Aqua	MODIS	95	6.0NRT	307.2	287.4	D
2	2	-36.918	146.782	336.7	1.0	1.0	2020-01-02	1520	Aqua	MODIS	100	6.0NRT	293.9	38.6	N
3	3	-16.985	138.283	343.4	1.2	1.1	2019-12-12	115	Terra	MODIS	85	6.0NRT	315.4	30.1	D
4	4	-14.865	131.262	311.5	1.5	1.2	2019-11-17	1335	Terra	MODIS	78	6.0NRT	300.1	11.6	N

After Removing less significant features and converting categorical value to numerical value:

	latitude	longitude	brightness	scan	track	acq_time	satellite	confidence	bright_t31	daynight	Year
0	-25.117	149.245	363.1	1.2	1.1	0	0	100	316.6	0	19
1	-32.263	123.294	349.3	3.4	1.7	500	1	95	307.2	0	20
2	-36.918	146.782	336.7	1.0	1.0	1520	1	100	293.9	1	20
3	-16.985	138.283	343.4	1.2	1.1	115	0	85	315.4	0	19
4	-14.865	131.262	311.5	1.5	1.2	1335	0	78	300.1	1	19

7. Climb the Leaderboard. Our best result has been obtained by using the polynomial kernel of degree 3 as described above in point 4. We observe that without basis functions, there are only 12 trainable parameters and the model severely underfits. Performance can be improved by increasing the no. of parameters to 144 using a polynomial kernel of degree 2. However, best performance is achieved for a polynomial kernel of degree 3 with 1728 parameters. Additionally, we have also observed that trying higher order polynomials of degree 4 and above results in performance degradation due to the model overfitting.