Project 4 Report: Regression Analysis

Ege Cetintas
Berkin Durmus
Shihan Gao
Due Date: 03/06/20

ECE 219 Winter 2020
Prof. Vwani Roychowdhury
UCLA, Department of ECE

Introduction

Regression analysis is a statistical procedure for estimating the relationship between a target variable and a set of describing features. In this project, we explore common practices for best performance of regression including linear regression, polynomial regression, neural network and random forest, along with pre-processing of datasets such as encoding, standardization and feature selection, as well as regularization and cross-validation in prevention of overfitting.

Data inspection

Here we have two sets of data imported, one is bike sharing dataset and one is video transcoding time dataset from online database. Firstly for the bike sharing dataset, there are 13 features such as temp, atemp, year, season, and 3 targets or labels considered including casual, registered, and cnt denoting the counts of users. The first step for data analysis is to take a look at the dataset concerning the correlation of features and the target variable, the distribution of the features, the categorical features. Secondly, for the video transcoding dataset including input and output video characteristics along with their time taken for different valid transcodings. There are 68784 data points each with 19 features such as duration, codec, height and width containing the information of the video, as well as o-codec, o-width and 0-height containing the information of the output for transcoding. The target variable considered is transcoding time.

Question 1: Pearson correlation matrix

Here we plot the heatmap of Pearson correlation matrix of dataset features. We import seaborn library for visualization and heatmap function specifically to plot rectangular data as a color-encoded matrix, and also pandas library to read the data file into Dataframe and then compute pairwise correlation of columns including features and targets with function corr using pearson method.

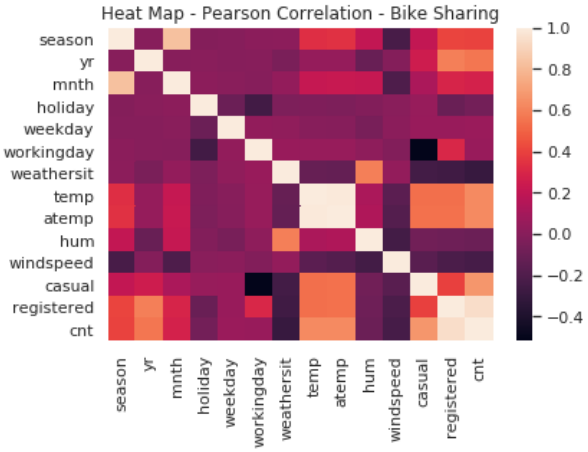


Figure 1: Pearson correlation matrix of bike sharing dataset

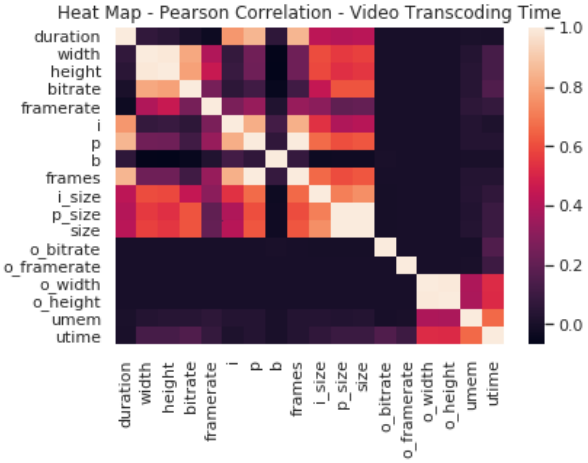
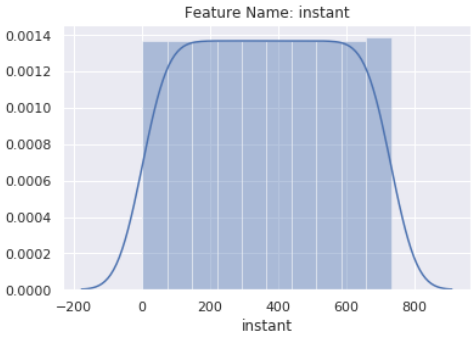


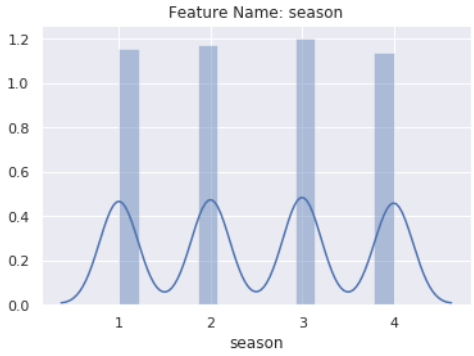
Figure 2: Pearson correlation matrix of video transcoding dataset

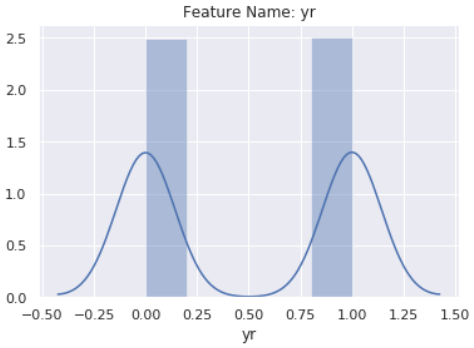
The color represents the pearson's correlation coefficient which returns a value of between -1 and +1 with higher absolute value meaning strong positive or negative correlation and 0 meaning no correlation.
For Bike Sharing dataset, 'temp', 'atemp', 'weathersit', 'year', 'season'seem to be correlated with our main target which is 'cnt' - total count. This is also expected as the temperature should be a limiting factor for bike users (cold weather might force them not to use bikes etc.). The 'season' appears in our list. This is also expected as it is also correlated with temperature.
For Video Transcoding Time dataset, 'o-width', 'o-heigt' and 'umen' seem to be correlated with our main target 'utime'. This makes sense as the output width and height in pixel used for transcoding should be correlated to total transcoding time for transcoding, and so is it for total codec allocated memory as 'umen'.

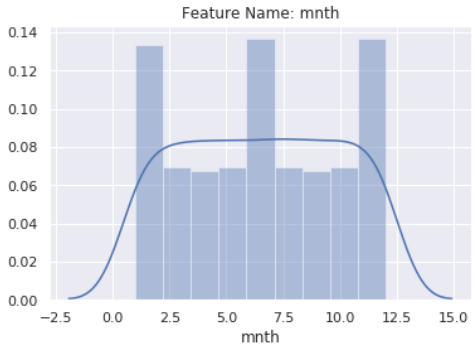
Question 2: Histogram of numerical features

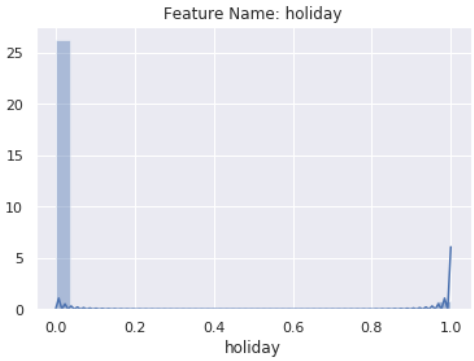
Here we plot the histogram of numerical features.

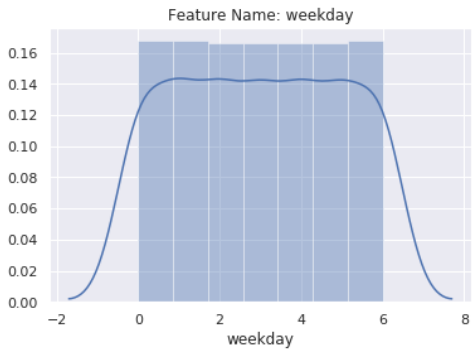


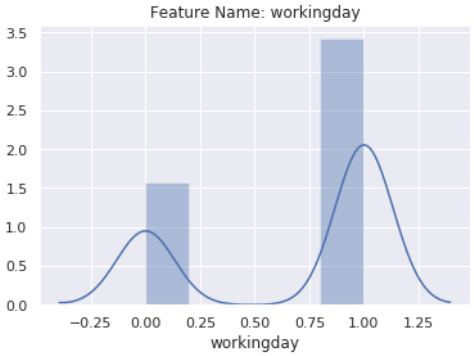


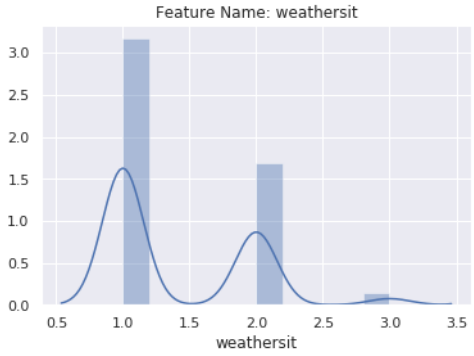


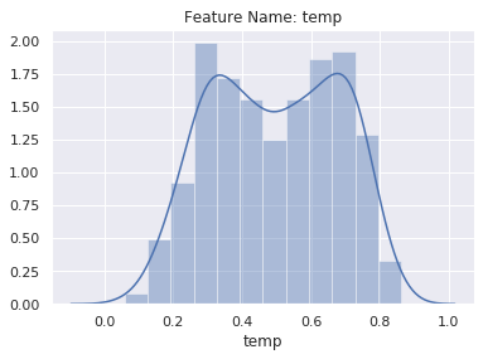


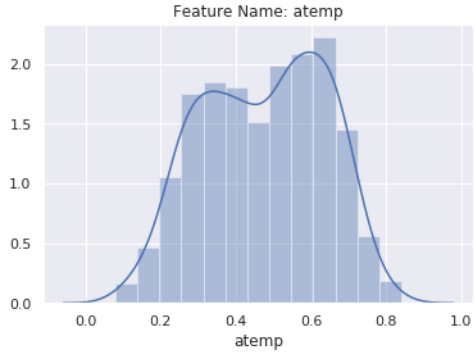


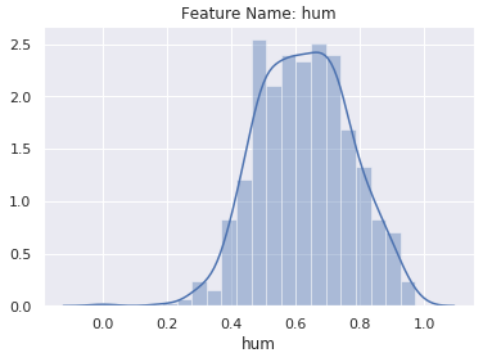


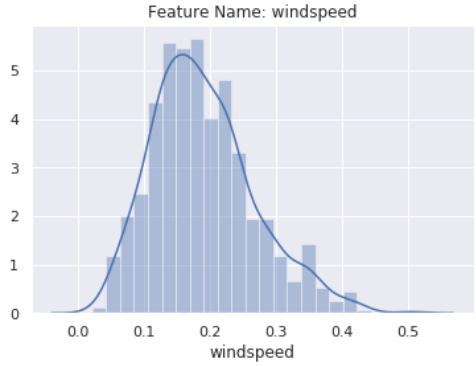


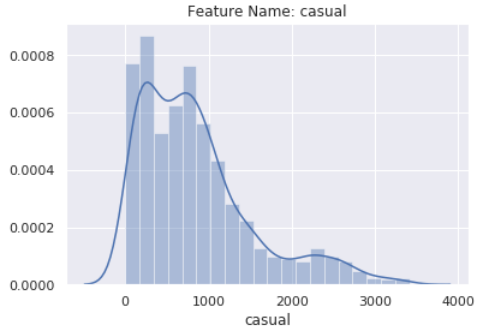


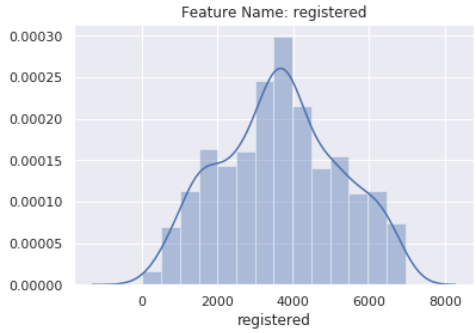


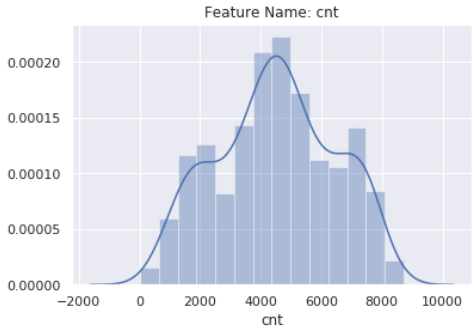






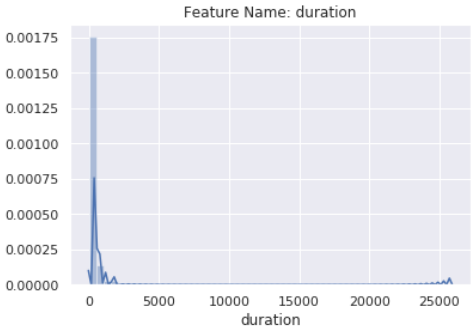


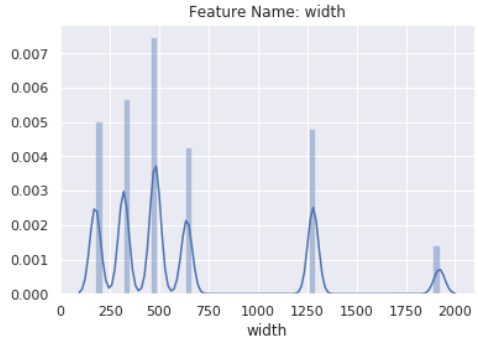


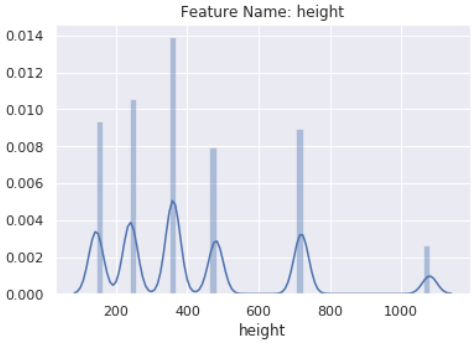


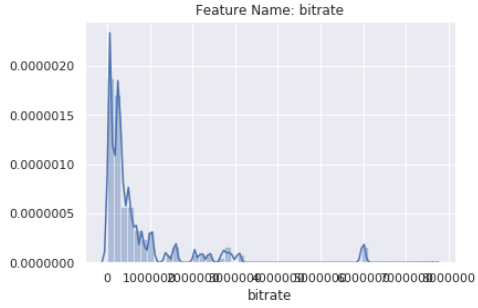
Here we can see that for some features the distribution has high skewness, such as holiday, temp and atemp which are also in bimodal shape or with nonnormal distribution,

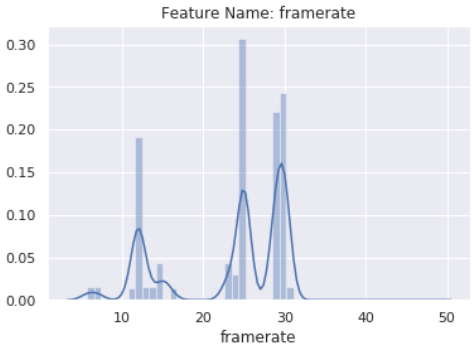
meaning that the data has come from two different systems. For features such as hum and windspeed the data is moderately skewed with normal distribution.

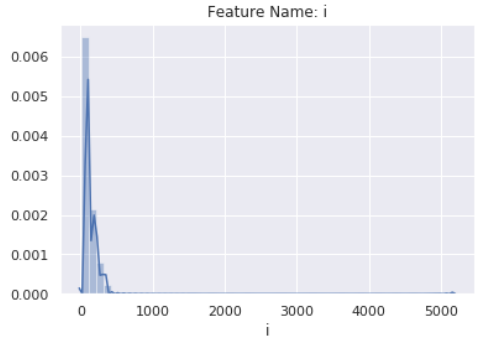


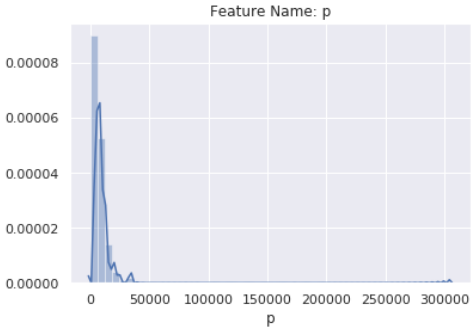


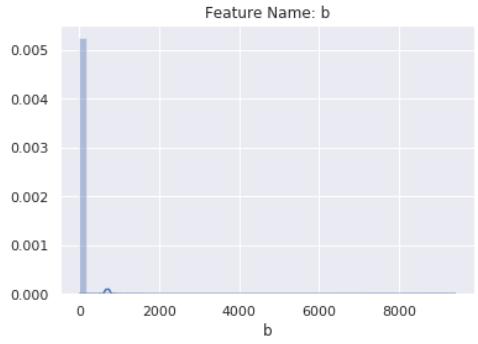


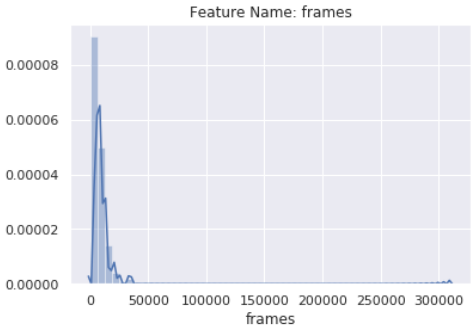


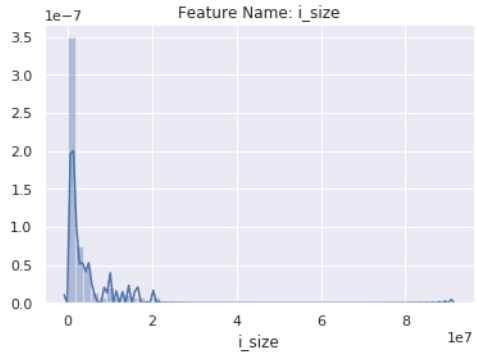


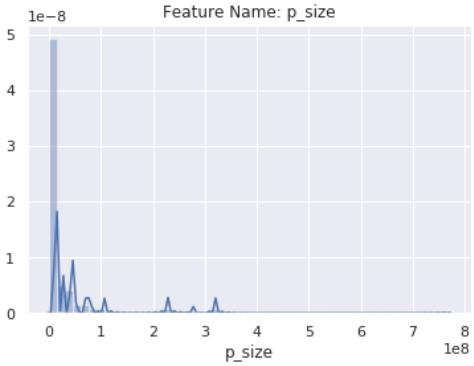


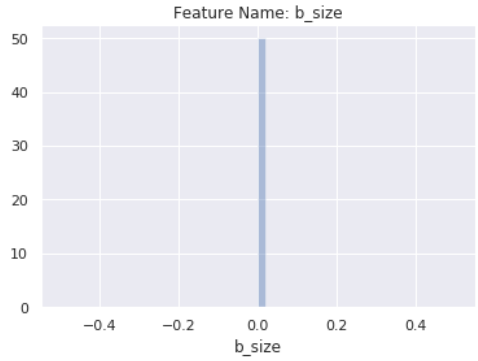


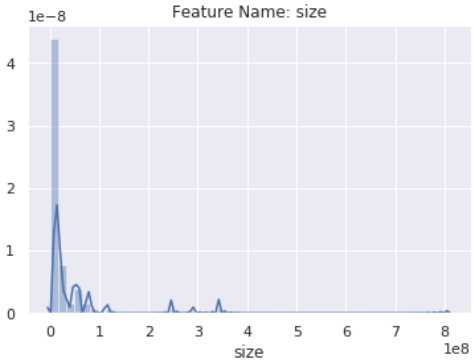


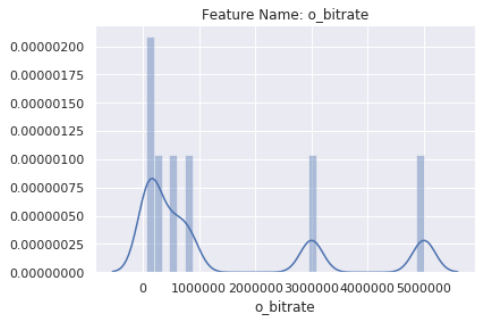

Feature Name: frames
0.00008
0.00006
0.00004
0.00002
0.00000
0
50000
100000
150000
200000
250000
300000
frames

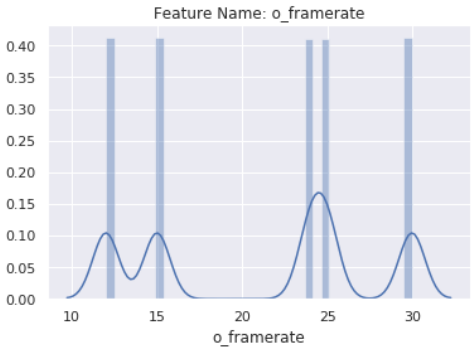

Feature Name: i_size
1e−7
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
2
4
6
8
1e7
i_size

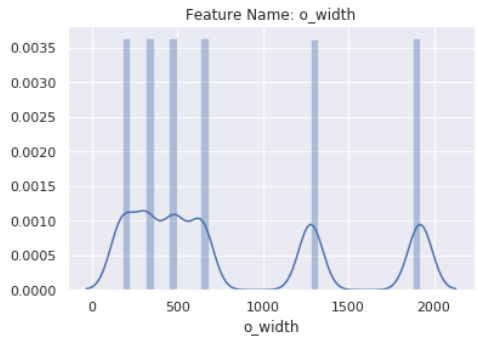

Feature Name: p_size
1e−8
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8
1e8
p_size

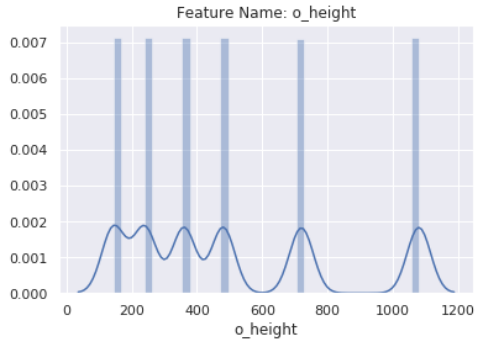

Feature Name: b_size
50
40
30
20
10
0
−0.4
−0.2
0.0
0.2
0.4
b_size

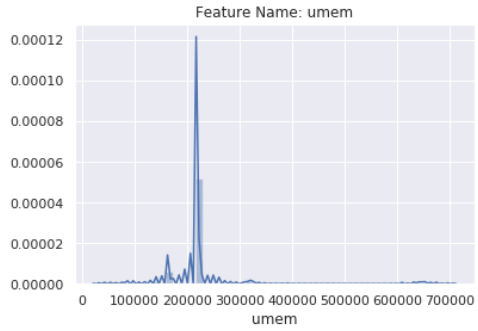

Feature Name: size
1e−8
4
3
2
1
0
0
1
2
3
4
5
6
7
8
1e8
size

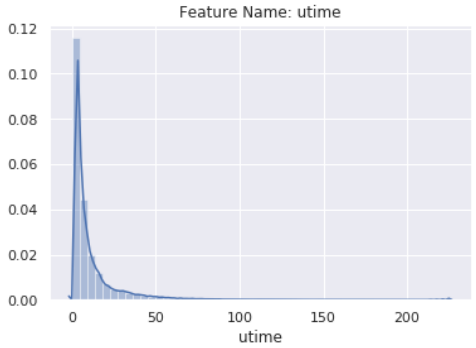

Feature Name: o_bitrate
0.00000200
0.00000175
0.00000150
0.00000125
0.00000100
0.00000075
0.00000050
0.00000025
0.00000000
0
1000000
2000000
3000000
4000000
5000000
o_bitrate


Feature Name: o_framerate
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
10
15
20
25
30
o_framerate


Feature Name: o_width
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
500
1000
1500
2000
o_width





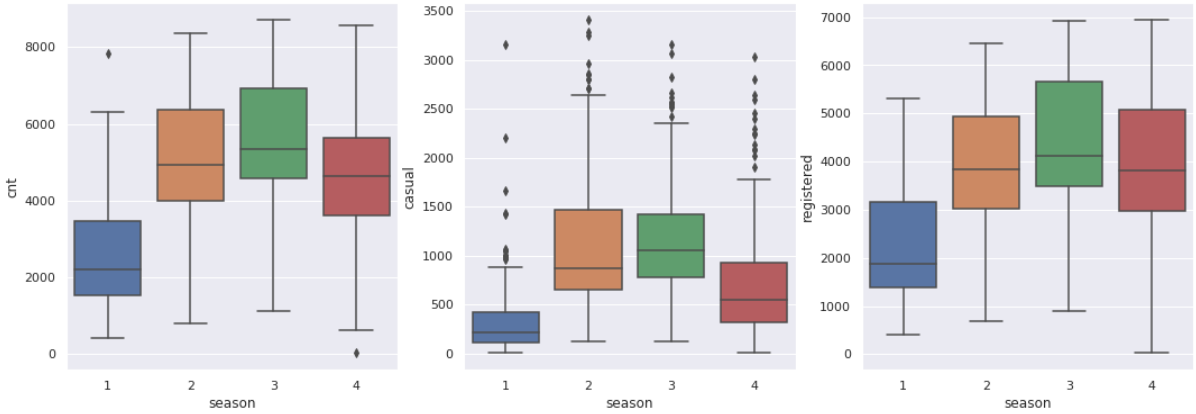


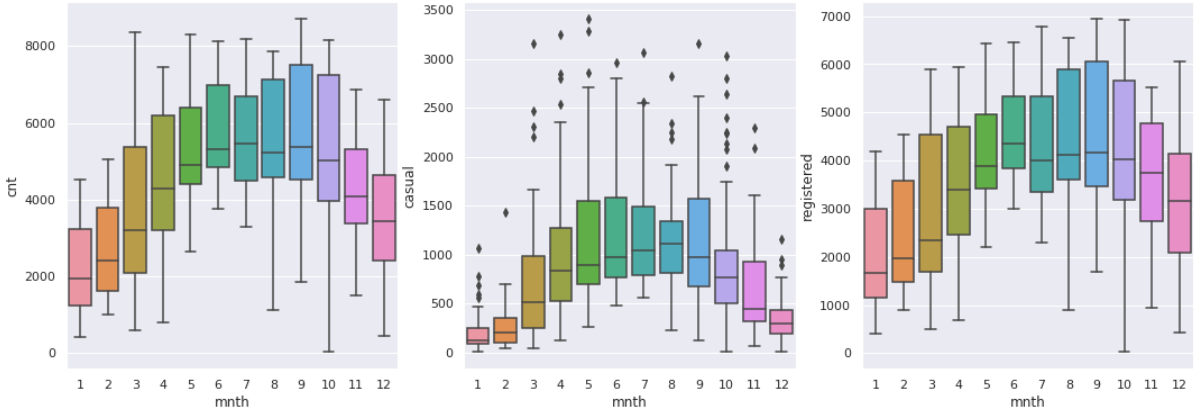
From the histogram of features of video transcoding dataset, we could see that features such as duration, bitrate, frames, as well number and size of different frames in the video have high skewness.

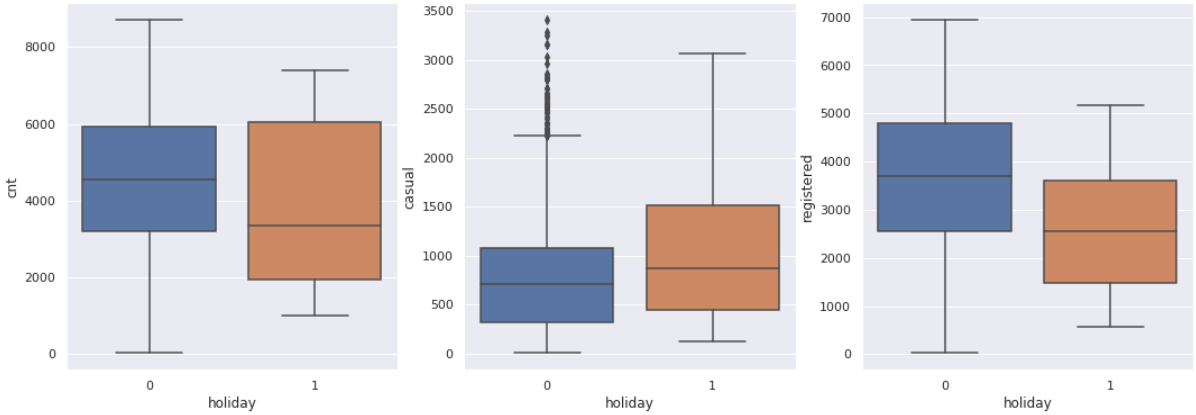
If the data has a skew or be shifted left or right, the data needs to be transformed prior to estimating the parameters, such as taking the log or square root, or more generally, using a power transform like the Box-Cox transform.

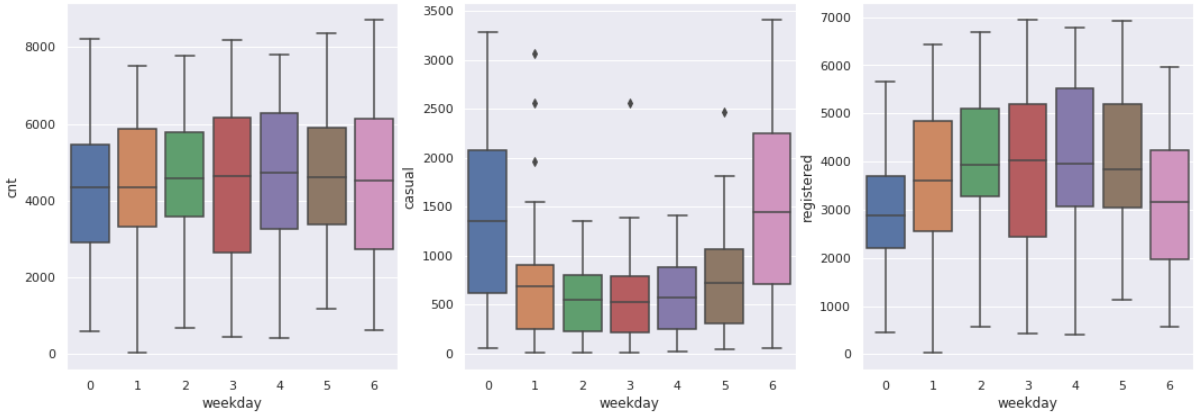
Question 3: Box plot of categorical features vs target variable

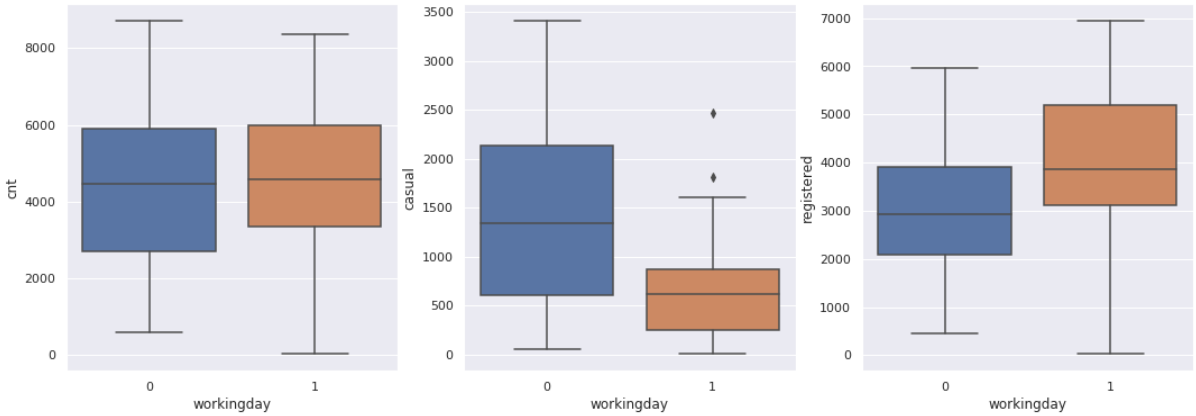
Here we draw vertical box plots of categorical features vs target variable to show distributions with respect to categories. The presented box of box plot shows the minimum, maximum, 1st quartile, 2nd quartile and 3rd quartile, while the whiskers extend to show the rest of the distribution. The 2nd quartile which is indicated by the the line inside the box represents the median value of the target or the label corresponding to different features, and the value ranges of 50 percents of the total number of labels around the median are indicated by the edge of the box. Outliers are shown by default beyond the edges of the whiskers. Thus from the box plots we can observe how the target data is distributed and spread for various feature categories.

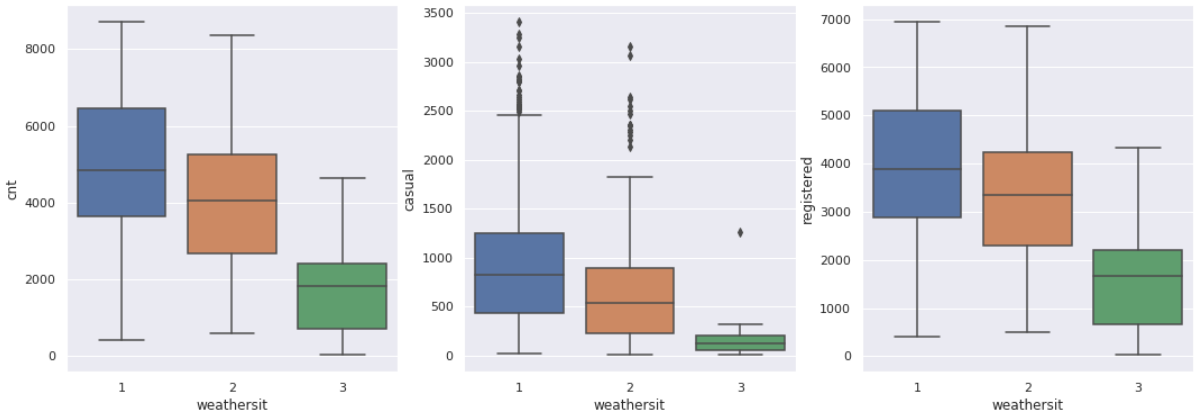

cnt
season
casual
season
registered
season


cnt
mnth
casual
mnth
registered
mnth


cnt
holiday
casual
holiday
registered
holiday

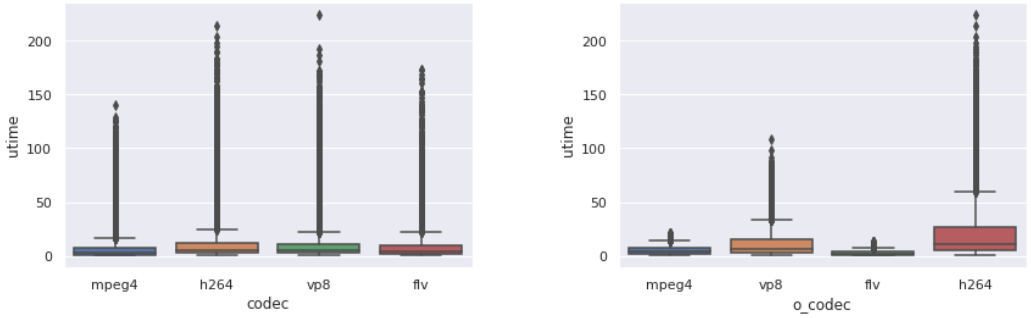






Take the feature season as an example, which can take on on of the limited number of possible values 1, 2, 3 and 4 and thus could be defined as a categorical feature. According to the box plot of target variables including casual, registered and cnt vs feature season, we can see that for different seasons the distribution of the counts of users changes a lot. Specifically for cnt which is the total number of rental bikes, most of the values are above 3500 in season 3, whereas most of the values are below 3200 in season 1. The height of the

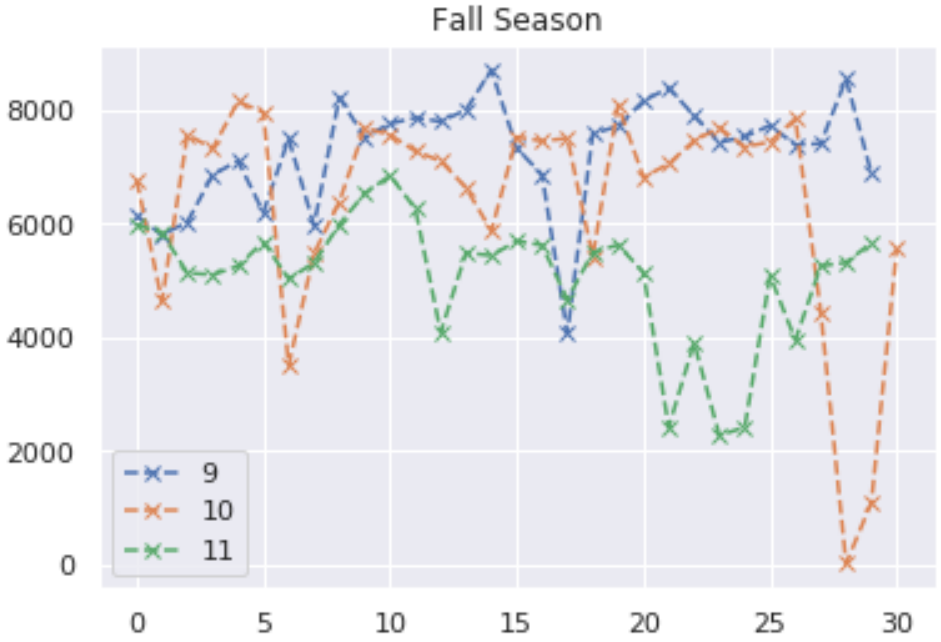
central rectangle along with the whiskers could approximately present the centralization of variance. A fat-tailed distribution is a probability distribution that exhibits a large skewness or kurtosis, which could be seen from the whisker. In the meantime, if the 2nd quartile line in the box deviates too much from the central position, the data distribution is not symmetric which also indicates higher skewness. In this case, for example, the counts of casual users are highly skewed in almost all seasons. In video transcoding dataset, take codec and o-codec features as an example.

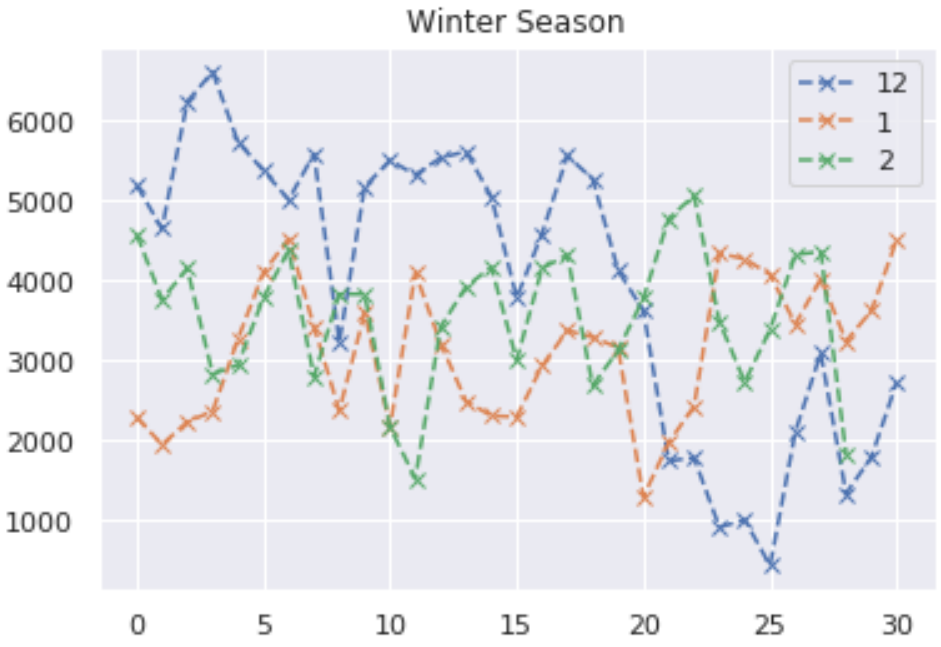


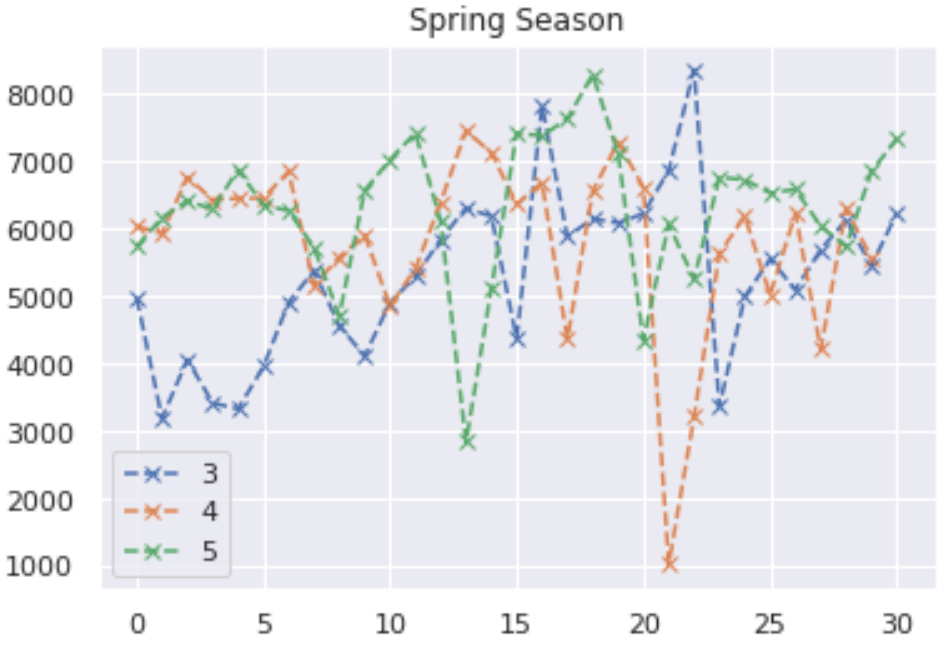
The plot indicates that the amount of outliers located outside the fences of the box-plot outside (1.5 times the interquartile range above the upper quartile and bellow the lower quartile) is very high which means these two features are far from being normally distributed and are highly skewed.

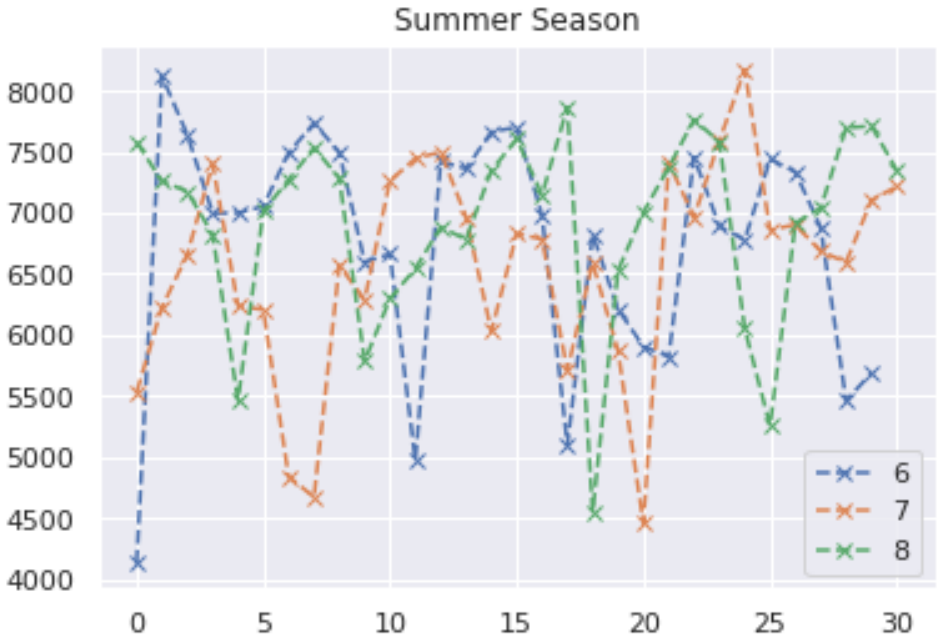
Question 4: Count number per days for months

For this question, we plotted the count number for every month yet grouped according to the season:




Winter Season
12
1
2
6000
5000
4000
3000
2000
1000
0
5
10
15
20
25
30


Spring Season
3
4
5
8000
7000
6000
5000
4000
3000
2000
1000
0
5
10
15
20
25
30



From these data, we can observe the winter months have a lower mean value of the count. If we have a look at the summer season, the months June and August have similar peaks and falls. This can be observed from the months April and May as well as January and February.

Question 5: Distribution of utime

Here we inspect the dataset by analyzing the distribution of the video transcoding times, which is the attribute "utime" in the data file.

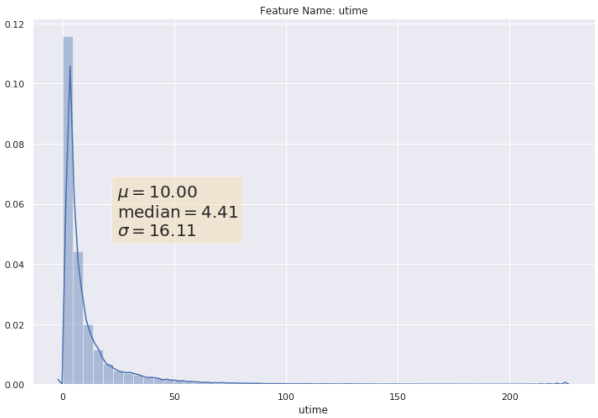


Figure 3: utime distribution

From the distribution plot we could see that the there is only one peak which means it's a unimodal distribution but with high positive skewness. The mean and median transcoding times are reported in the figure.

Data preprocessing

A categorical feature is a feature that can take on one of a limited number of possible values. A preprocessing step is to convert categorical variables into numbers and thus prepared for training.

Question 6: Encoding of categorical features

Categorical data are variables that contain finite label values each representing a different category. Some categories may have a natural relationship to each other, such as a natural ordering such as season, year, month and weekday. There are two ways of encoding categorical features considered, one is ont-hot and another is the scalar. One-hot is to encode the categorical data to binary vectors denoting the presence or absence of the feature. Each row has one feature with a value of 1, and the other features with value 0. The number of vectors depends on the number of categories for features. This method produces a lot of columns that slows down the learning significantly if the number of the category is very high for the feature. Although one-hot encoding works appropriately for nominal data, it discards the numeric relationship between each category such as ordinal.

A scalar encoder encodes a numeric (floating point) value into a bit map SDR (Sparse Distributed Representation) with a given number bits of possible bits, of which width bits are 1 and the rest are 0. One important property of the SDR is that values close together share many more bits than values further apart thus it contains the relationship between each category. However, the encoding is linear which means that the neighboring bucket indices must overlap by a linearly decreasing number of bits. Thus it works properly to find a linear relationship between input feature values and outputs but no more complex relationship.
Here we use "get-dummies" to convert categorical variable into dummy or indicator variables for one-hot encoding, and then then dummy-encoded columns are concatenated to the original feature values.

Question 7: Standardization of feature columns

Standardization of datasets is a common requirement for many machine learning estimators. Datasets might behave badly if the individual features do not more-or-less look like standard normally distributed data which follows Gaussian with zero mean and unit variance. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.
Here we standardize feature data by removing the mean and scaling to unit variance.

Question 8: Feature selection

We use mutual information (MI) between two random variables to measure the dependency between the variables. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency. Here we import the mutual info regression which relies on nonparametric methods based on entropy estimation from

k-nearest neighbors distances. We use F scores to compare the significance of the improvement of a model with respect to the addition of new variables. Here we import the f regression which is a linear model for testing the individual effect of each of many regressors, where the correlation between each regressor and the target is computed and then converted to an F score then to a p-value.
An F statistic is a value in a regression analysis to find out if the means between two populations are significantly different. $F_{stat} = MSB/MSW$, where MSB and MSW are mean square between and within variables. The F-score is a uni-variate feature selection method, which means that it scores each of the features individually (higher score is better), without considering that a feature may improve in combination with another feature. A higher F-score reflects a better predictive model.

feature	MI	F score
season	0.21560329	143.96765259
yr	0.2747256	344.89058554
mnth	0.37674401	62.00462455
holiday	0.01147239	3.42144104
weekday	0.04429331	3.33109137
workingday	0.02353521	2.73674228
weathersit	0.06291107	70.72929783
temp	0.38823587	473.47171053
atemp	0.46458243	482.45431053
hum	0.0452059	7.46194
windspeed	0.05501197	42.4378415

Training

Once the data is prepared, we would like to train multiple algorithms and compare their performance using RMSE.

Linear regression

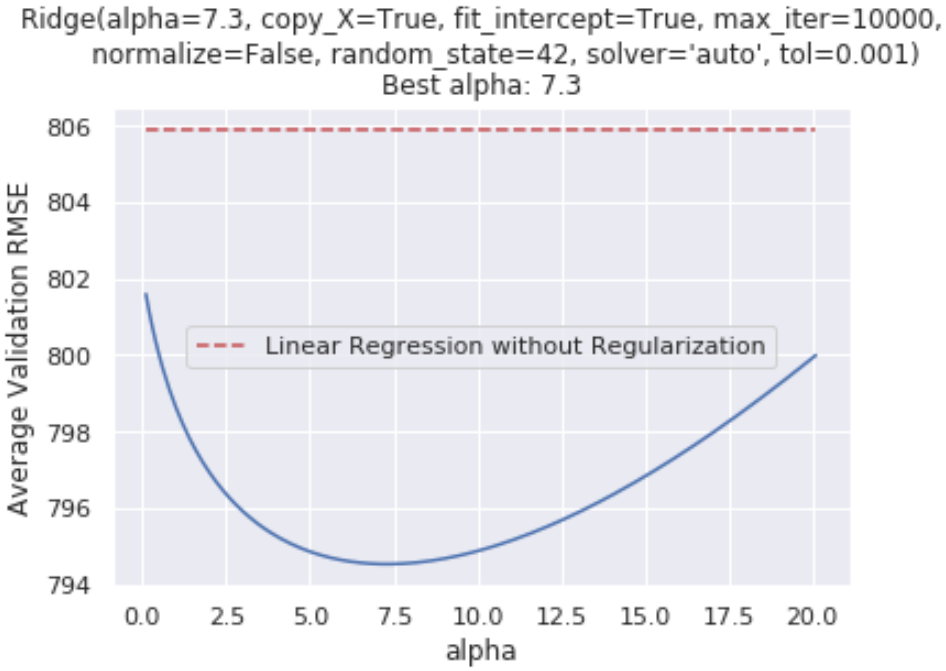
Question 9: Regularization explanation

Lasso and Ridge regression are two linear regression models to reduce model complexity and prevent over-fitting which may result from simple linear regression. Lasso applies L1 regularization and ridge applies L2 regularization. In lasso regression, L1 regularizer can lead to zero coefficients i.e. some of the features are completely neglected for the evaluation of output. So lasso regression not only helps in reducing over-fitting but it can help us in feature selection. In ridge regression, L2 regularizer shrinks the coefficients and it helps to reduce the model complexity and multi-collinearity for hypothesis.

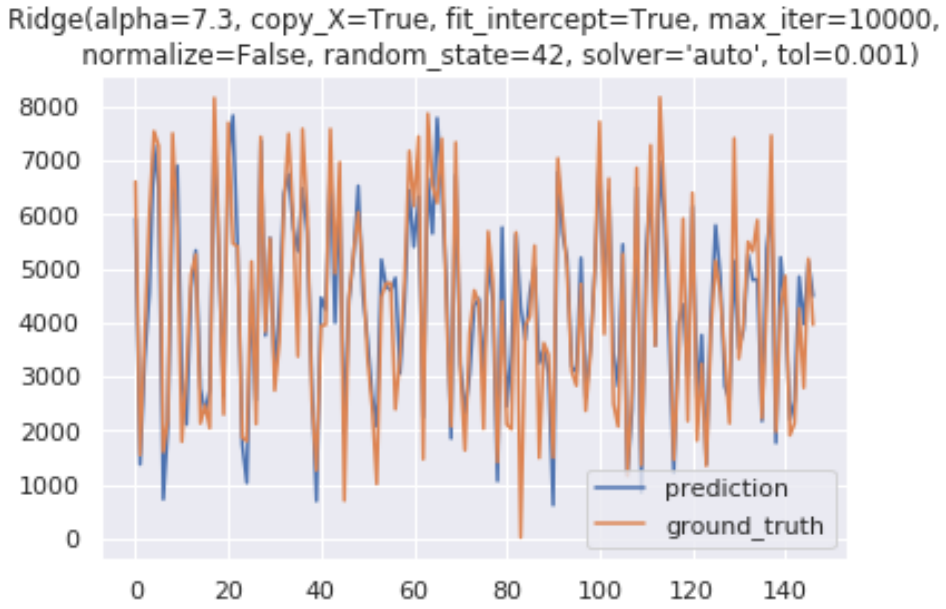
Question 10: Choosing best regularization scheme

In the first part of the question, we report the results generated with the bike-sharing dataset. Later, we report the results from the transcoding time dataset.
Regularization improves the conditioning of the problem and reduces the variance of

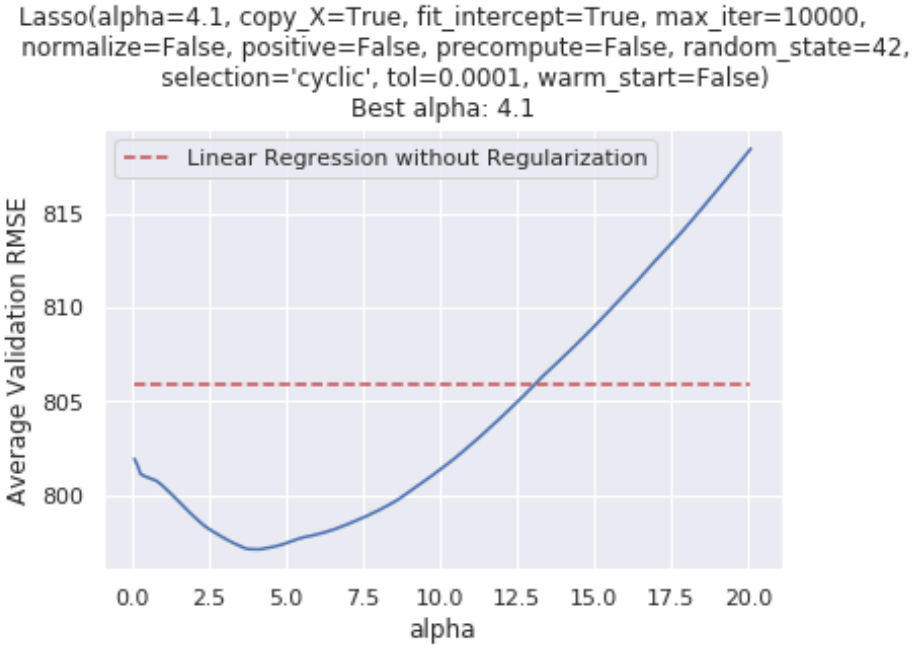
the estimates thus reduces the test error. Here we plot RMSE of test dataset using k-folds cross-validation (k=10) vs regularization strength alpha for linear regression model without regularization, Lasso and Ridge. Larger alpha values specify stronger regularization. Firstly, for ridge regression, we first plot RMSE vs alpha to determine the best regularization parameter.



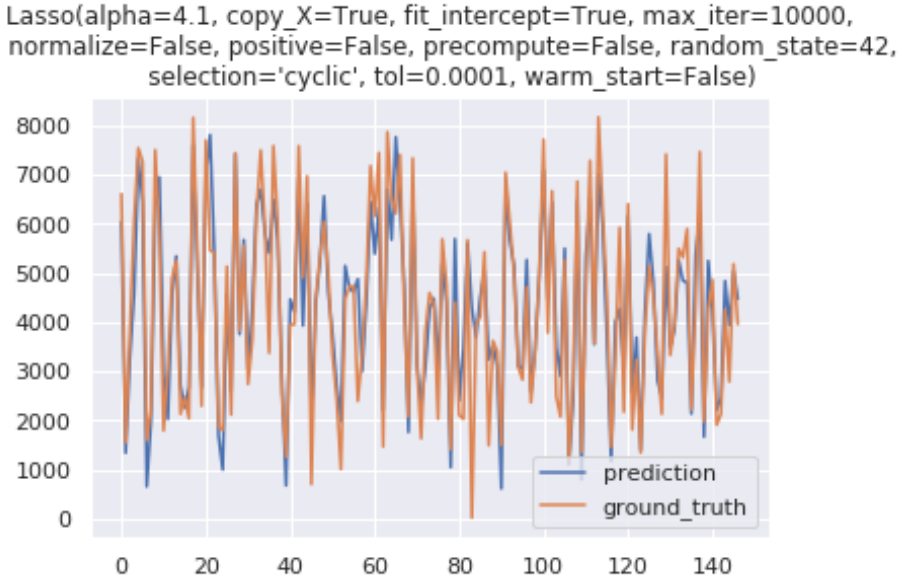
This plot indicates that the best alpha is 7.3 corresponding to RMSE of test set 827.25, which is then used to predict labels in comparison with the ground true target values.



The plot above visually shows good prediction of ridge regression.
Secondly, for lasso regression, we first plot RMSE vs alpha to determine the best regularization parameter.

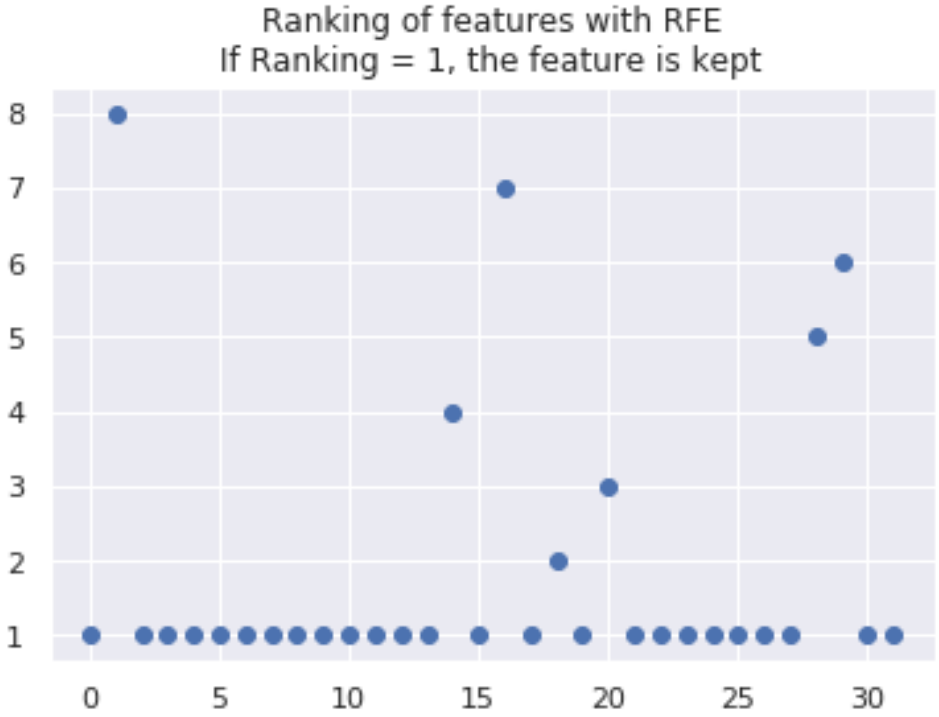


This plot indicates that the best alpha is 4.1 corresponding to RMSE of test set 829.48, which is then used to predict labels in comparison with the ground true target values.



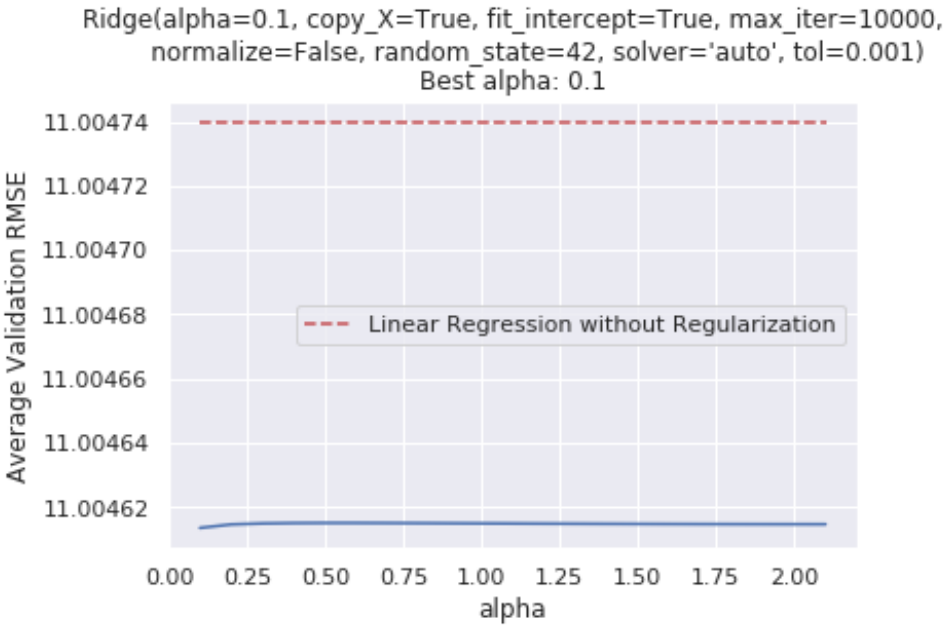
The plot above visually shows good prediction of lasso regression.
We use recursive feature elimination for feature selection which could help reduce overfitting and training time, and improve accuracy. Sklearn provides RFE for recursive feature

elimination and RFECV for finding the ranks and here we chose number of features to be 25. RFE is to select features by recursively considering smaller and smaller sets of features according to the weights, and the least important features are pruned from current set of features. That procedure is recursively repeated on the pruned set until the number of features to select is achieved, which is 25 in this case.

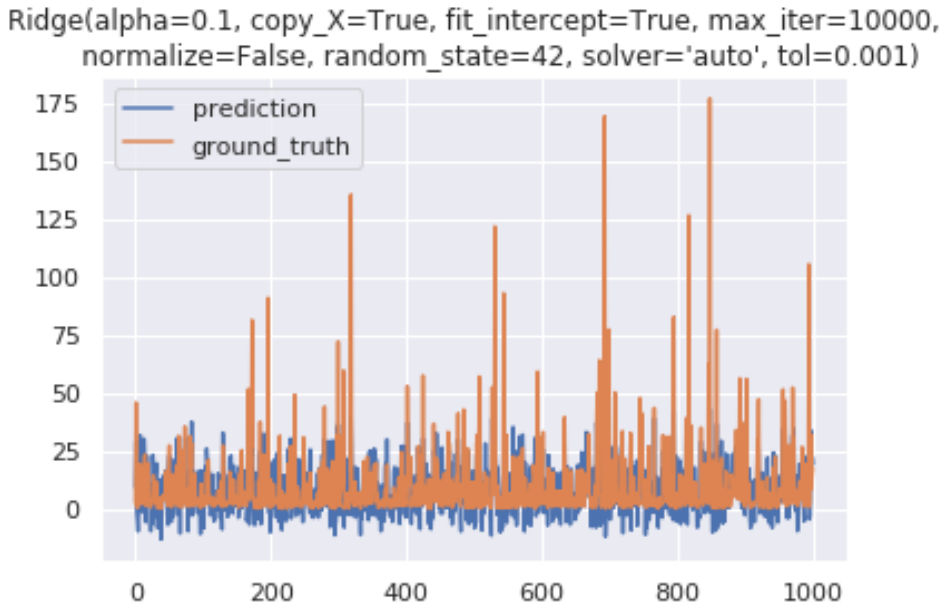


After feature selection, the test set RMSE is reduced to 828.24.
Consequently, the best regularization scheme in this case is ridge regression with the optimal penalty paramater as 7.3.

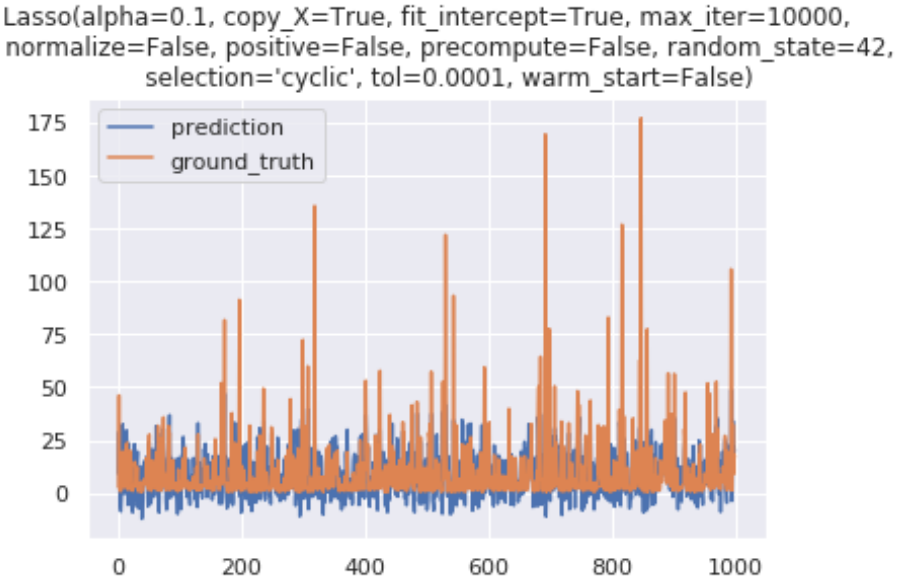
For video transcoding dataset, we explore the regularization scheme effect on the learned hypothesis and firstly with ridge regression RMSE for the test dataset is plotted as below.



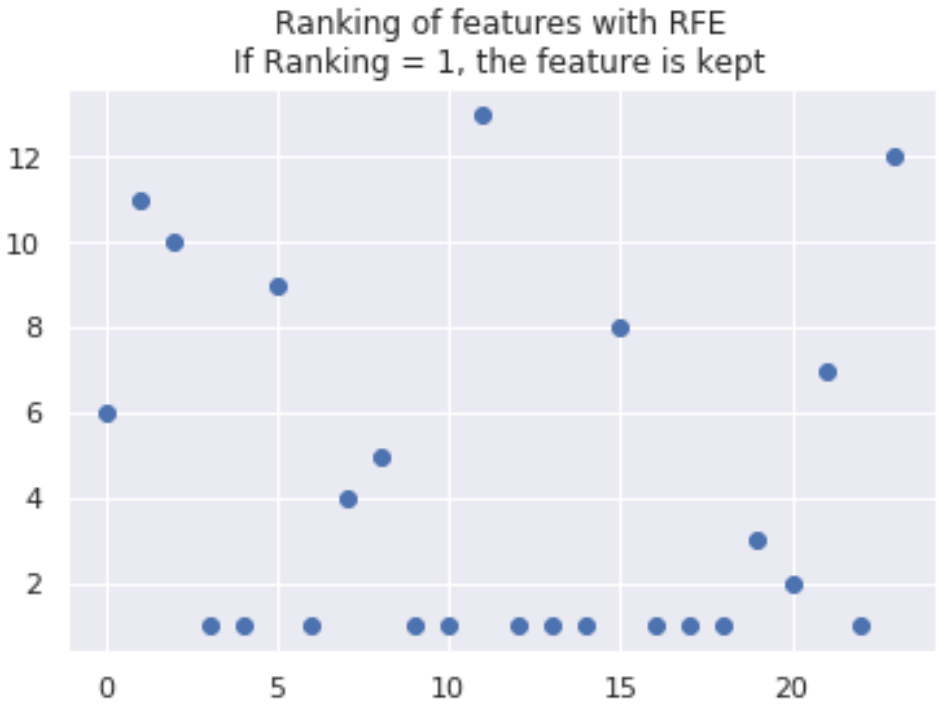
Linear ridge regression RMSE is reduced to 10.97 in comparison with no regularization as 11.00. With the increase of alpha, the performance only changes slightly within small values and keep approximately constant then, which indicates no significant either overfitting or underfitting.



However, for lasso regression model, RMSE is larger than in linear regression without regularization, and keeps increasing with rising alpha value which is due to underfitting. In this case, regularization does not improve the performance of model.

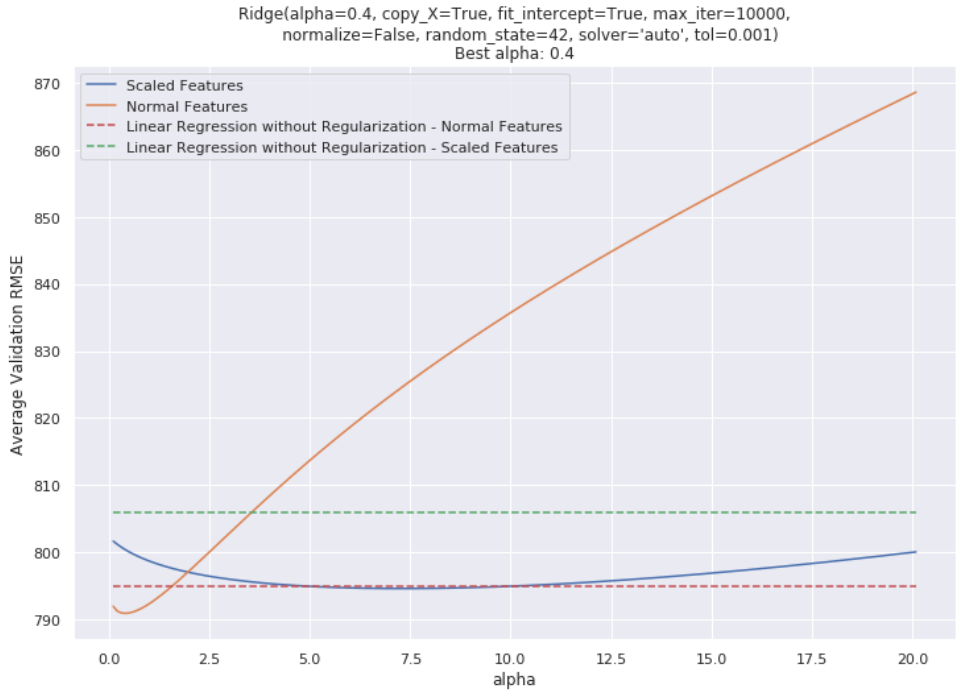


Then we use RFE to rank features for selection again. Test set RMSE is 10.97 which is not very different with the value before selection.

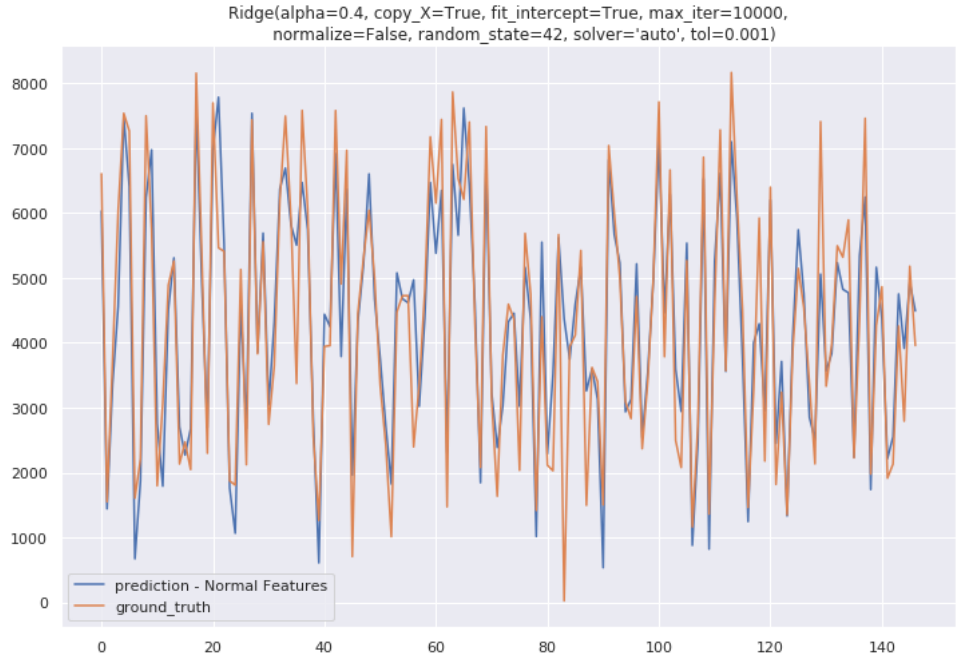


Question 11: Effects of feature scaling

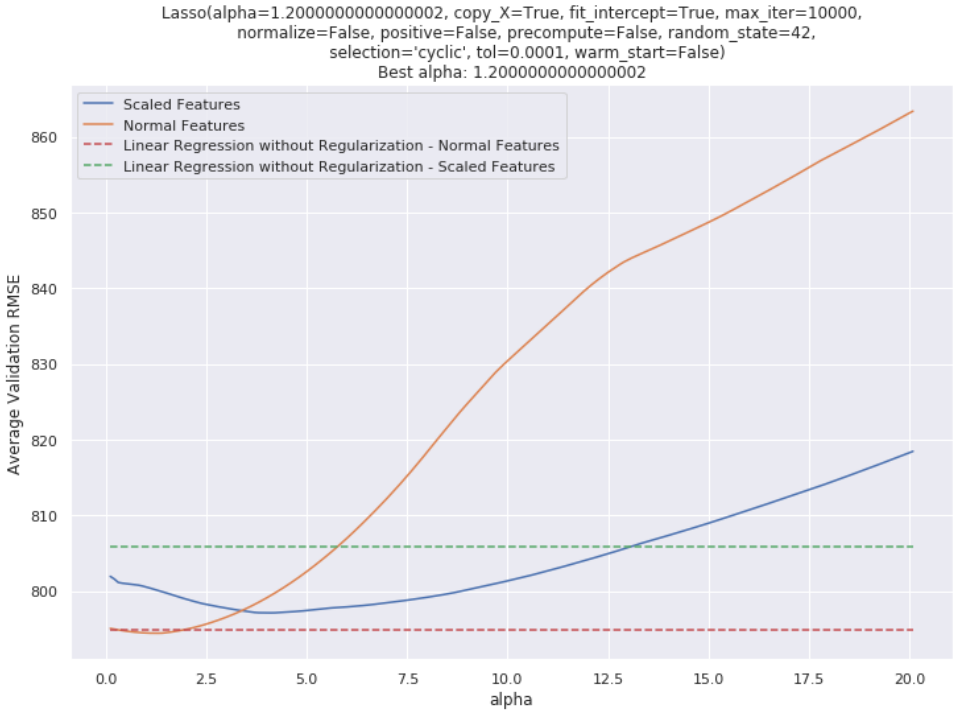
Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. Standardization is widely used as a preprocessing step in many learning algorithms to rescale the features to zero-mean and unit-variance. For regression models like Lasso and Ridge, scaled features with regularization it will give better result than no scaling with regularization. The reason is these models get affected by outliers very severely and if features have wide range of values with outliers, RMSE will be high for test dataset.
Firstly, for ridge regression, we plot RMSE of linear regression with and without regularization.



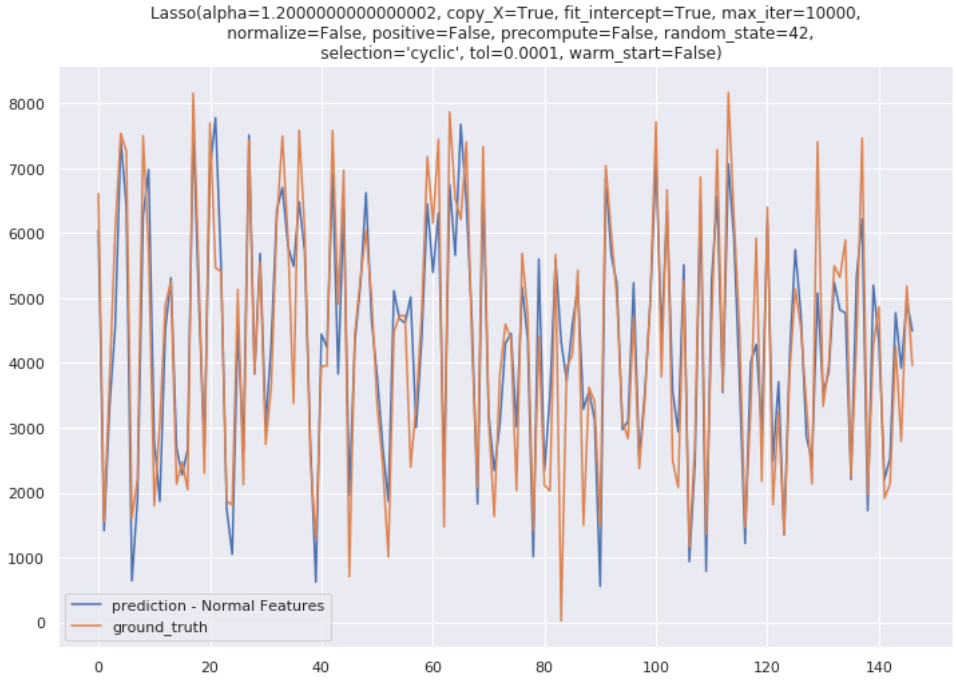
The plot indicates that RMSE cannot be reduced through solely feature scaling. For reference, the RMSE of linear regression with unscaling features and without regularization is reported to be 794.82. However, the combination of scaling features and regularization does not improve the performance of the algorithm necessarily for small small penalty (less than about 2.0), and standardization of features could change the optimal penalty parameter to 0.4, in this case. Then with alpha=0.4, we plot the prediction in comparison with the ground truth values.



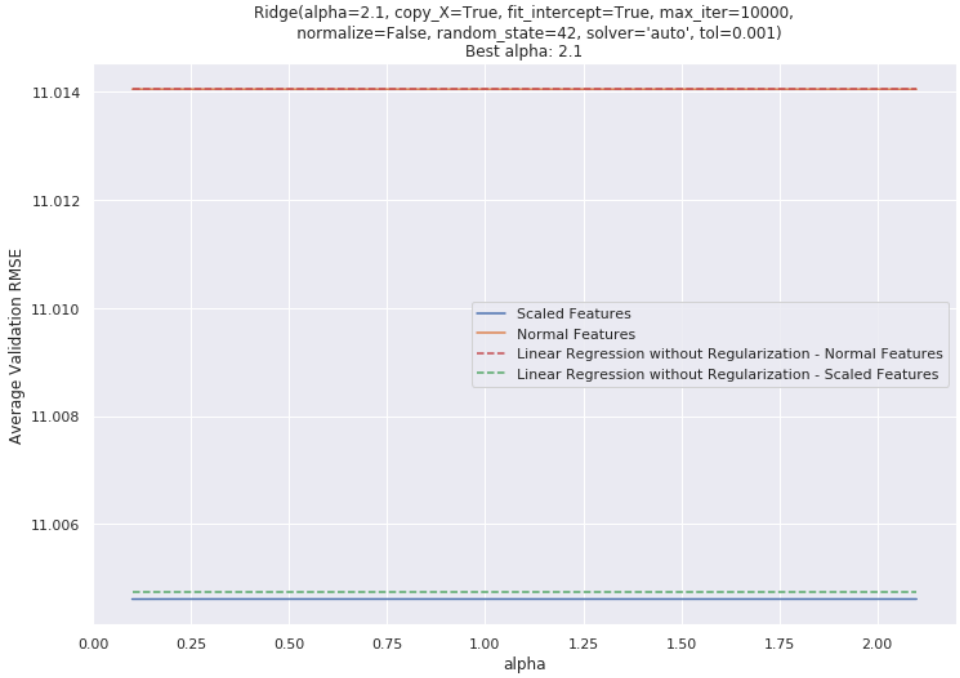
Secondly, for lasso regression, we plot RMSE of linear regression with and without regularization.

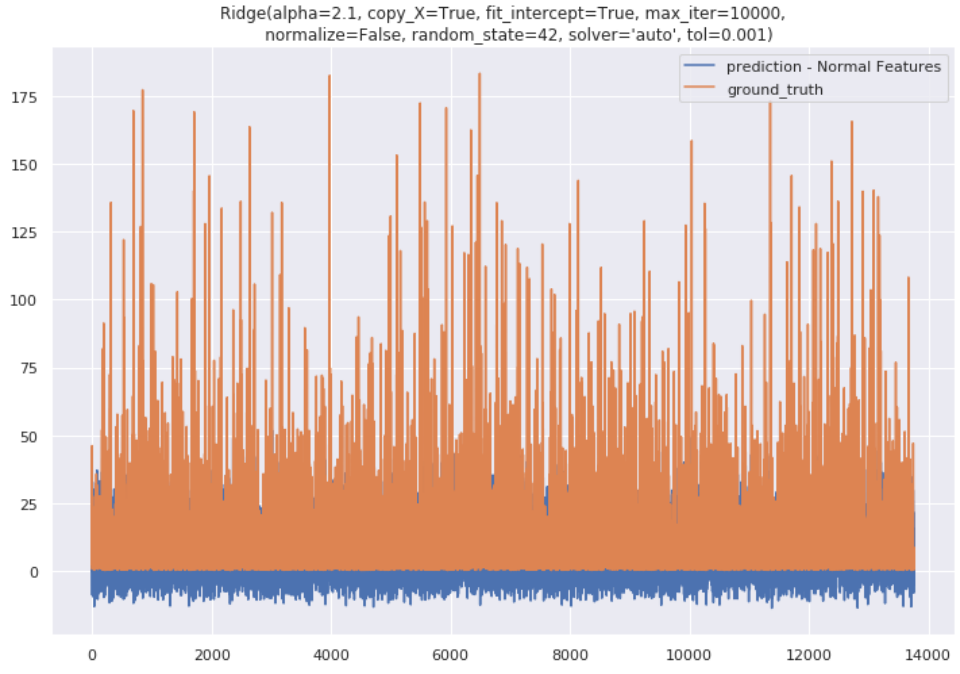


In this case, the optimal penalty parameter of lasso regression is changed to 1.2. Similarly, scaling feature it self does not reduce RMSE but could improve the performance of algorithm on test dataset in combination with optimal regularizer.



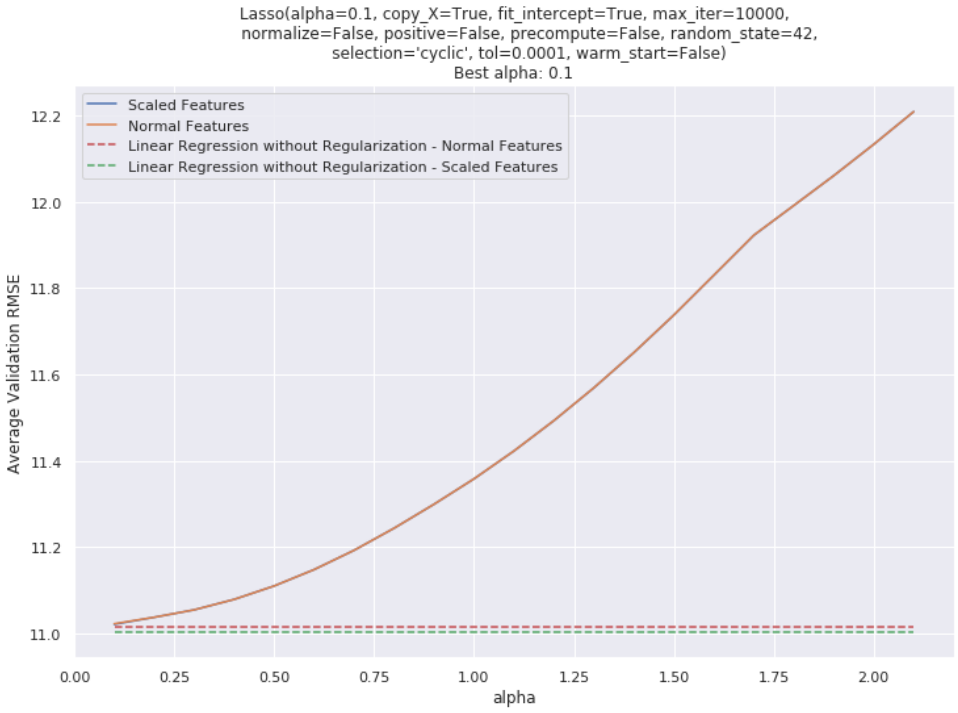
Next, for the video transcoding dataset we report the performance of models with different penalty parameters with and without feature scaling.

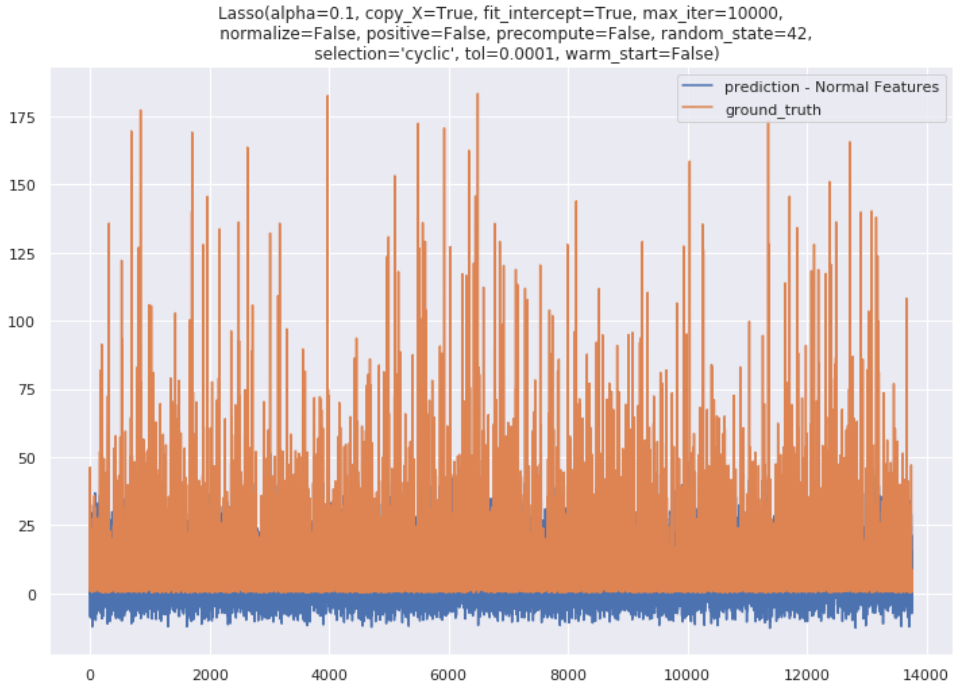




For video transcoding dataset, standardization helps reduce RMSE of validation dataset. However, there is not much difference with regularization especially for non-scaling features. For reference, linear regression without regularization is reported to be

11.01. Test set RMSE is reduced to be 10.97. Then for comparison, we plot our prediction and ground truth value accordingly.
In lasso regression model, with increasing of alpha, average validation RMSE keeps increasing which indicates continuous underfitting problem as shown below.





In the case of no regularization, RMSE with normal features is slightly lower than with scaled features which again proves no improvement with just feature scaling.

Question 12: p-values of features

The p-value for each term tests the null hypothesis that the weight is equal to zero which meaning no significance given to the feature. A low p-value (¡ 0.05) indicates the null hypothesis should be rejected. In other words, a predictor that has a low p-value is likely to be a meaningful addition to your model because changes in the predictor's value are related to changes in the response variable.
As an example, below, we plot the p-values vs the features that was found in one of our linear regression models:

	coef	std err	t	P>|t|
const	2282.7484	16.433	138.916	0.000
holiday	-68.7322	36.077	-1.905	0.057
workingday	61.7068	28.542	2.162	0.031
weathersit	-352.0803	42.591	-8.267	0.000
temp	635.7394	270.510	2.350	0.019
atemp	256.5312	250.434	1.024	0.306
hum	-197.0470	46.576	-4.231	0.000
windspeed	-208.2184	36.156	-5.759	0.000
season_1	-275.5448	141.867	-1.942	0.053
season_2	650.0689	137.592	4.725	0.000
season_3	508.2358	138.583	3.667	0.000
season_4	1399.9886	142.759	9.807	0.000
yr_0	141.0388	34.335	4.108	0.000
yr_1	2141.7096	33.754	63.451	0.000
mnth_1	-8.9442	203.400	-0.044	0.965
mnth_2	56.6677	194.894	0.291	0.771
mnth_3	566.0673	153.367	3.691	0.000
mnth_4	286.7030	174.712	1.641	0.101
mnth_5	559.0110	179.270	3.118	0.002
mnth_6	317.5336	171.286	1.854	0.064
mnth_7	-292.8109	202.420	-1.447	0.149
mnth_8	181.4214	194.380	0.933	0.351
mnth_9	822.9585	152.221	5.406	0.000
mnth_10	309.4834	177.769	1.741	0.082
mnth_11	-281.7171	193.539	-1.456	0.146
mnth_12	-233.6253	173.081	-1.350	0.178
weekday_0	82.3540	58.962	1.397	0.163
weekday_1	250.7787	82.539	3.038	0.002
weekday_2	262.2174	77.618	3.378	0.001
weekday_3	348.6008	79.130	4.405	0.000
weekday_4	339.1648	75.298	4.504	0.000
weekday_5	443.4232	80.223	5.527	0.000
weekday_6	556.2096	59.242	9.389	0.000

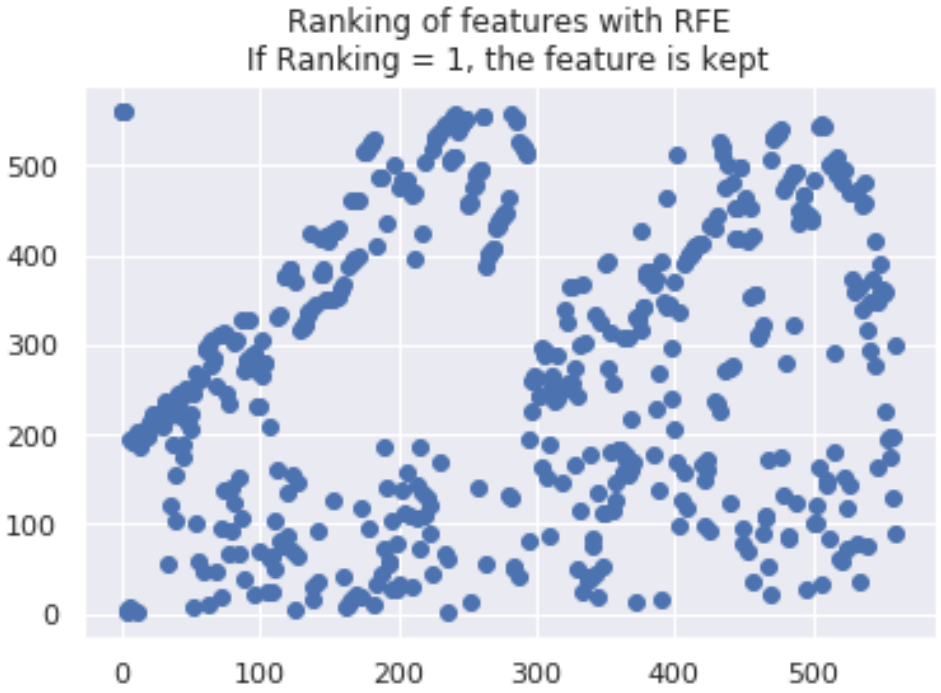
We can observe that 'weathersit' and 'temp' has low p-values hence they are relatively important features of our model.

Polynomial regression

Perform polynomial regression by crafting products of raw features up to a certain degree and applying linear regression on the compound features.

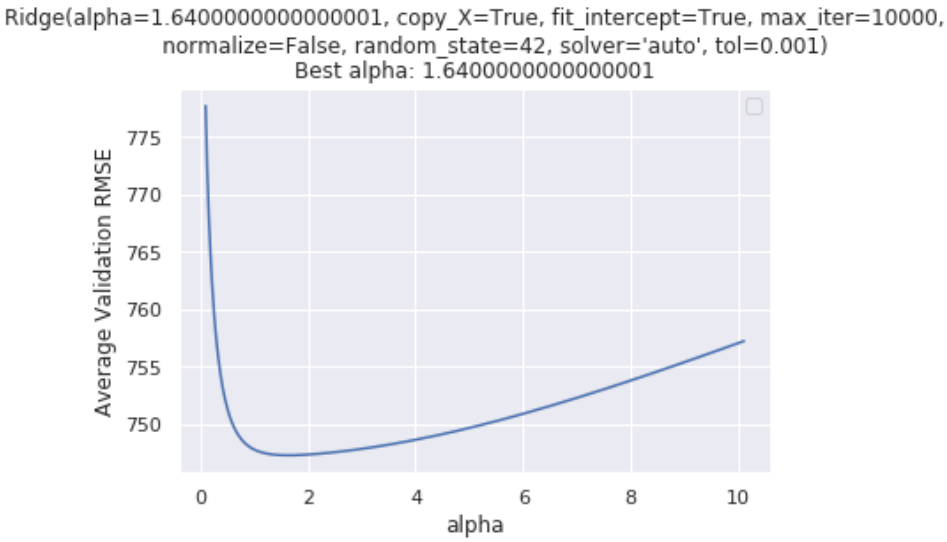
Question 13: Salient features

Here we use recursive feature elimination to rank the features and select the most salient ones. A low ranking value indicates a good feature and here we prune the predictors with ranking above 1.

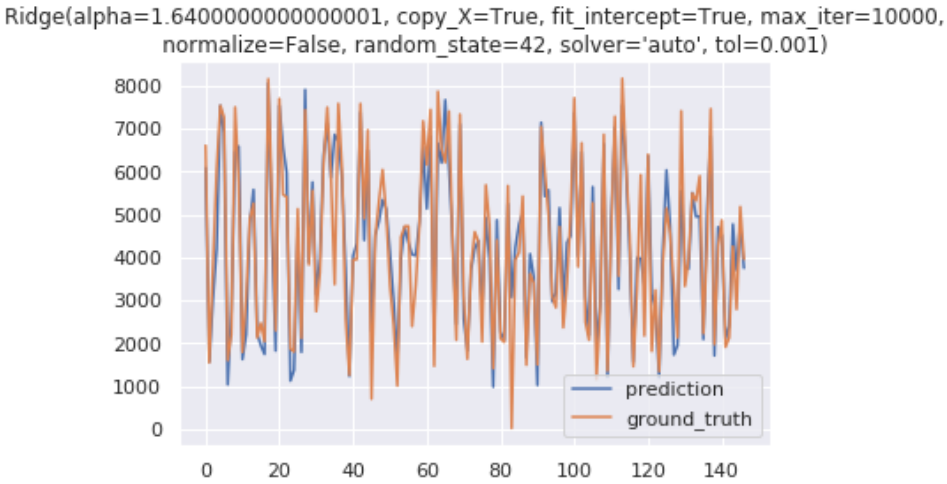


Question 14: Polynomial degree

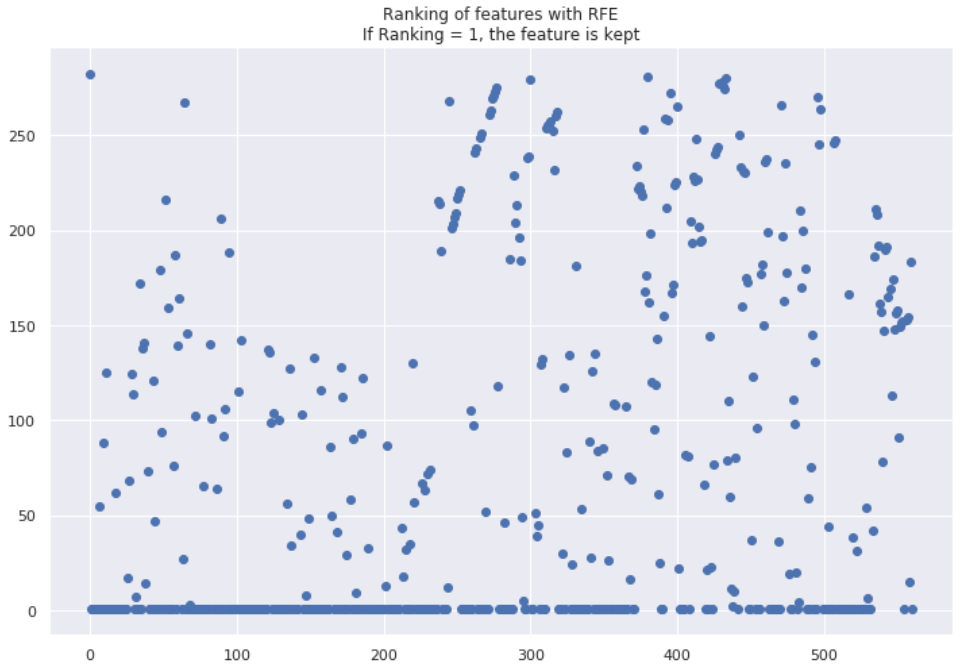
Here we set the degree to be 2 and plot the average validation RMSE vs the regularization penalty parameter for ridge regression. Firstly, for bike sharing dataset, the results are shown as below.



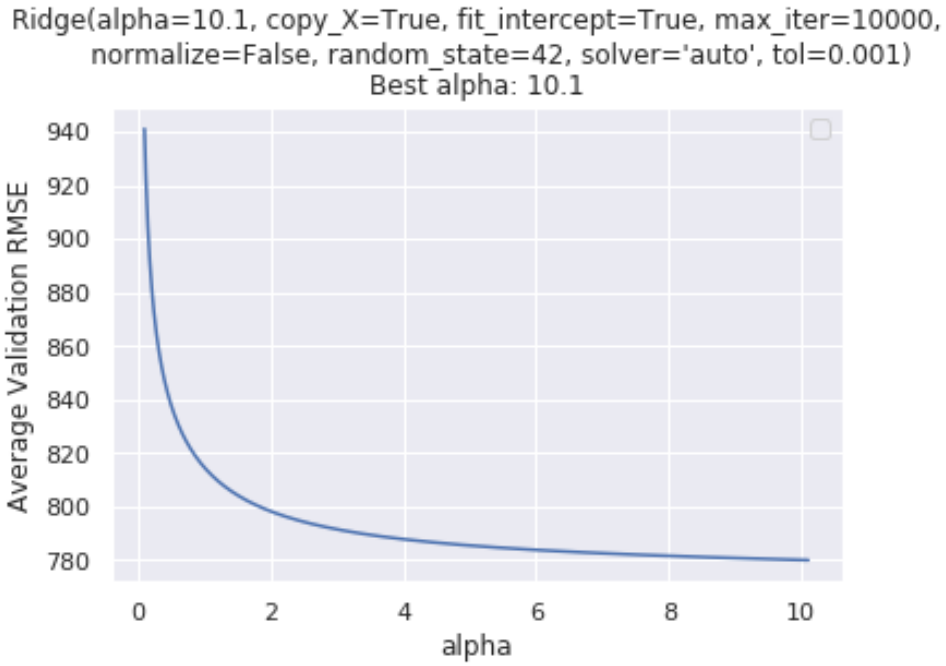
The optimal penalty parameter is reported to be 1.64 corresponding to the validation set RMSE as 736.50. However, underfitting occurs after alpha increases more than 1.64. Using too large a value of aplha can cause your hypothesis to underfit the data. A large value of results in large regularization penalty and thus a strong preference for simpler models which can underfit the data.The comparative curve of prediction of the best model and the true label are plotted.



Then, we rank the feature with RFE and keep the features with rank of 1.

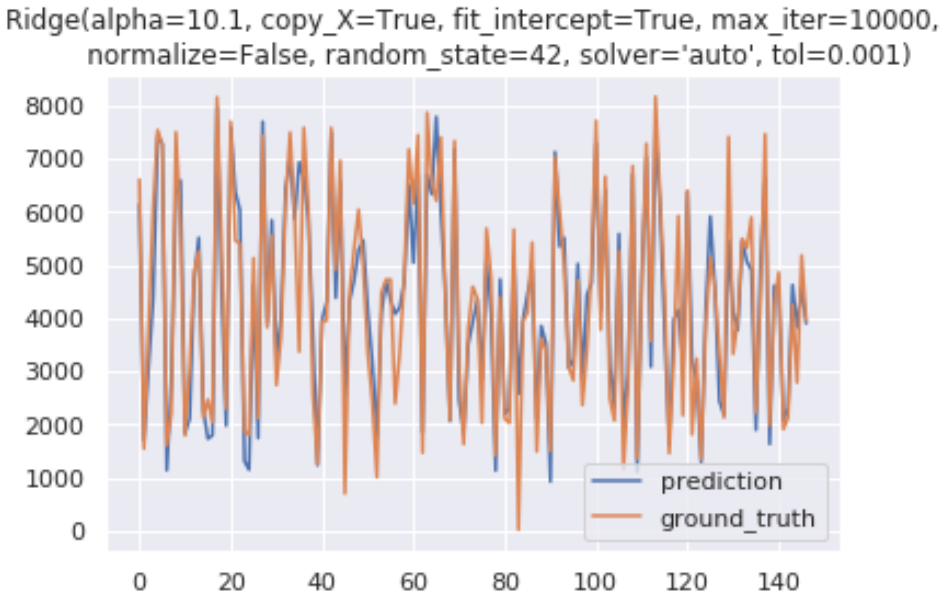


Here we use the optimal penalty parameter to calculate the validation set RMSE after feature selection. RMSE is reported to be reduced to 734.09.
Here we set the degree to be 3 and plot the average validation RMSE vs the regularization penalty parameter for ridge regression.

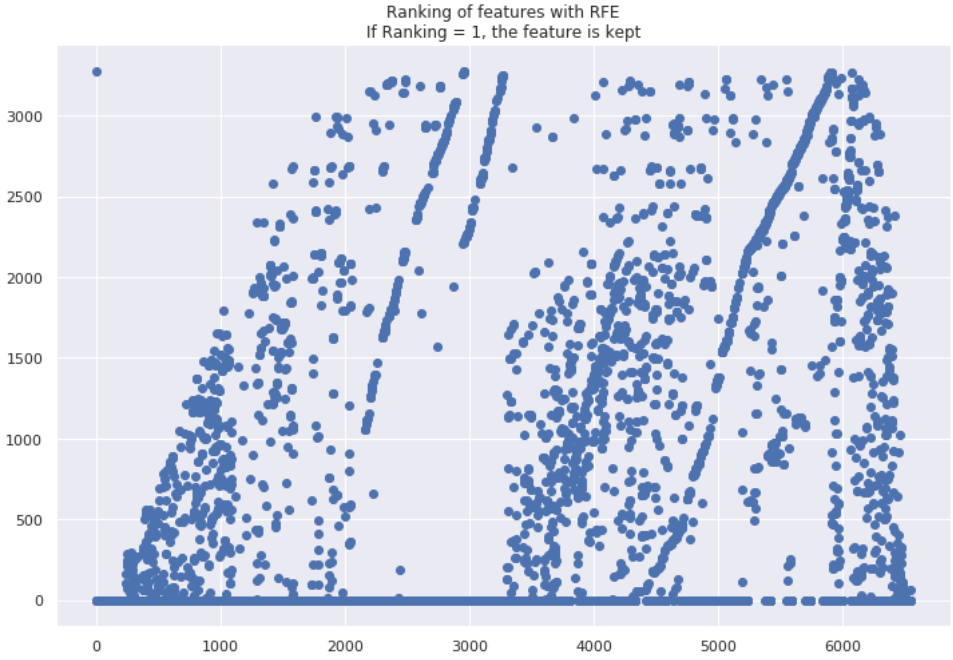


The optimal penalty parameter is reported to be 10 corresponding to the validation set RMSE as 732.87. However, there is no overfitting issue in this case, the average validation

RMSE keeps decreasing with rising alpha. The comparative curve of prediction of the best model and the true label are plotted.



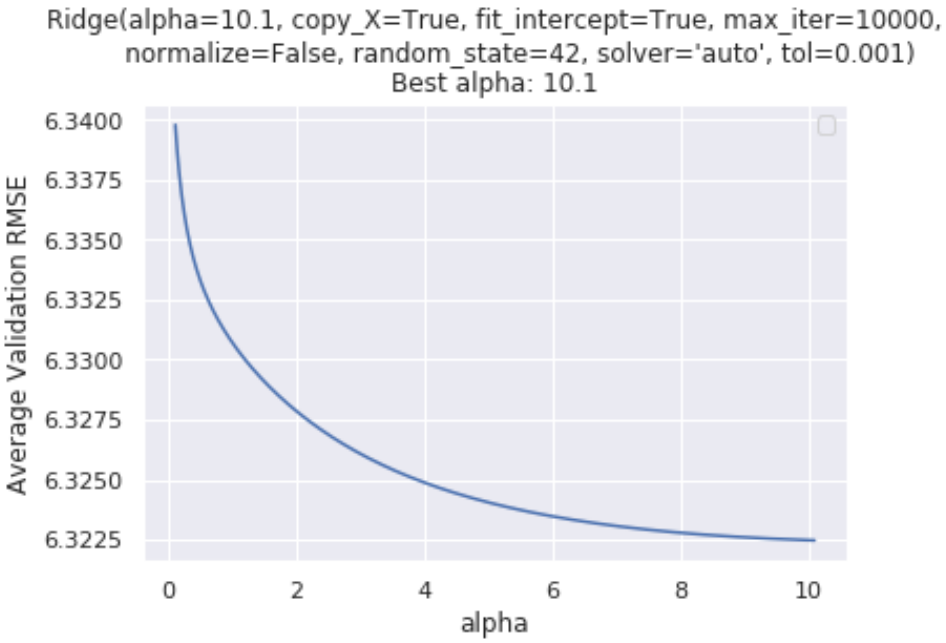
Then, we rank the feature with RFE and keep the features with rank of 1.



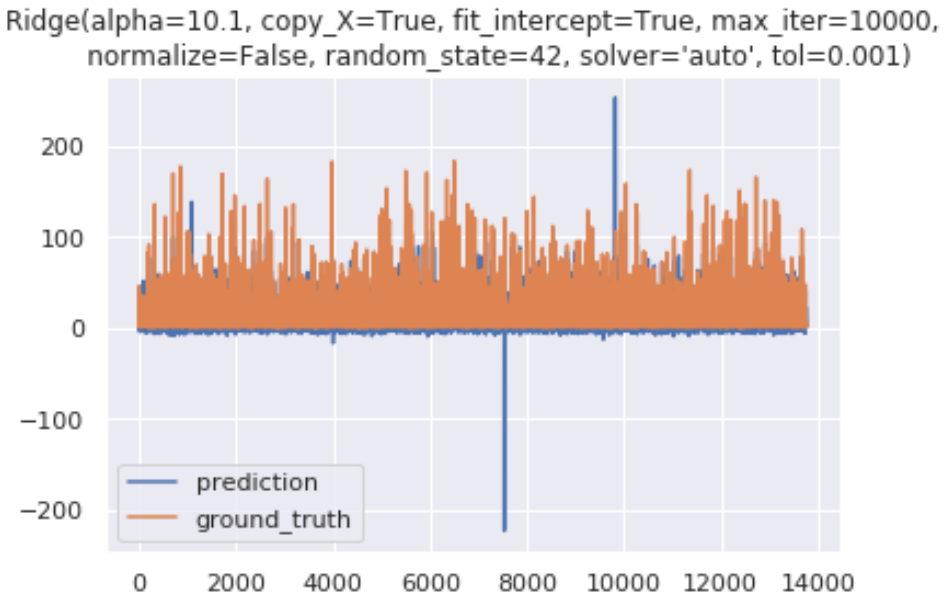
Here we use the optimal penalty parameter to calculate the validation set RMSE after feature selection. RMSE is not changed due to feature selection.

Overall, polynomial regression with degree as 3 predicts better than 2 in consideration of RMSE for the validation dataset. However, too much increase of the polynomial degree could cause the problem of overfitting because of the high variance of the estimators, which could explain the reason of the continuous decreasing trend of RMSE for degree 3 polynomial ridge regression. To prevent the potential overfitting problem, the regularization penalty parameter needs to increase correspondingly and we do not have underfitting problem relative to large alpha in this case.

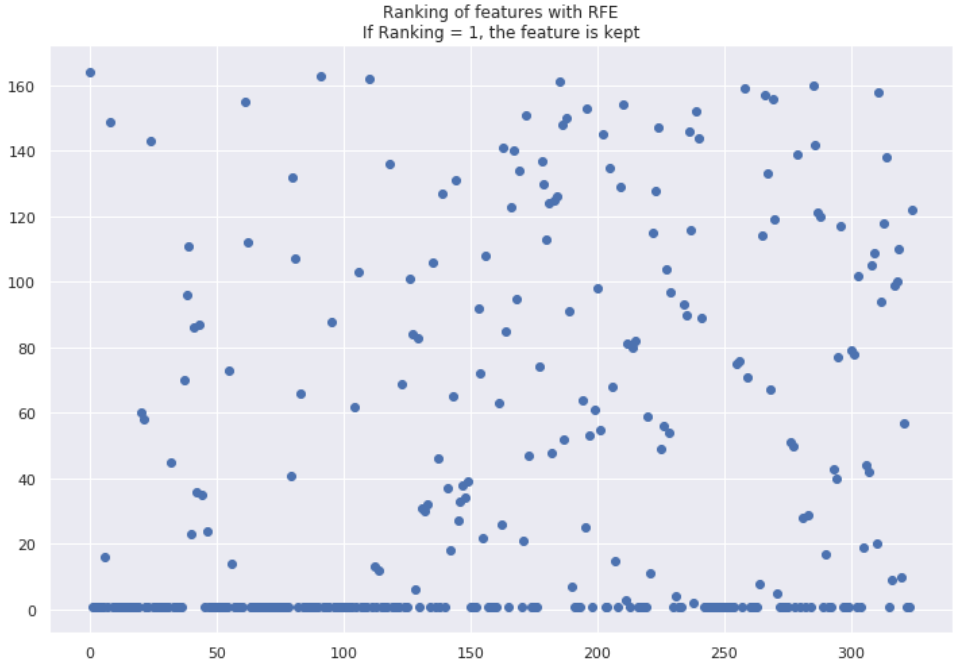
For video transcoding dataset, we still choose ridge regression model and the results are shown as below for degree 2. However, due to large computational cost, we could not implement the regression model with degree as 3.



RMSE keeps decreasing with rising alpha which implies mitigation effect of higher penalty parameter on potential overfitting introduced the raised power of inputs.



Then, we rank the feature with RFE and keep the features with rank of 1.



Here we use the optimal penalty parameter to calculate the validation set RMSE after feature selection. RMSE is reduced from 7.01 before selection to 6.88 after selection.

Question 15: Crafting inverse of certain features

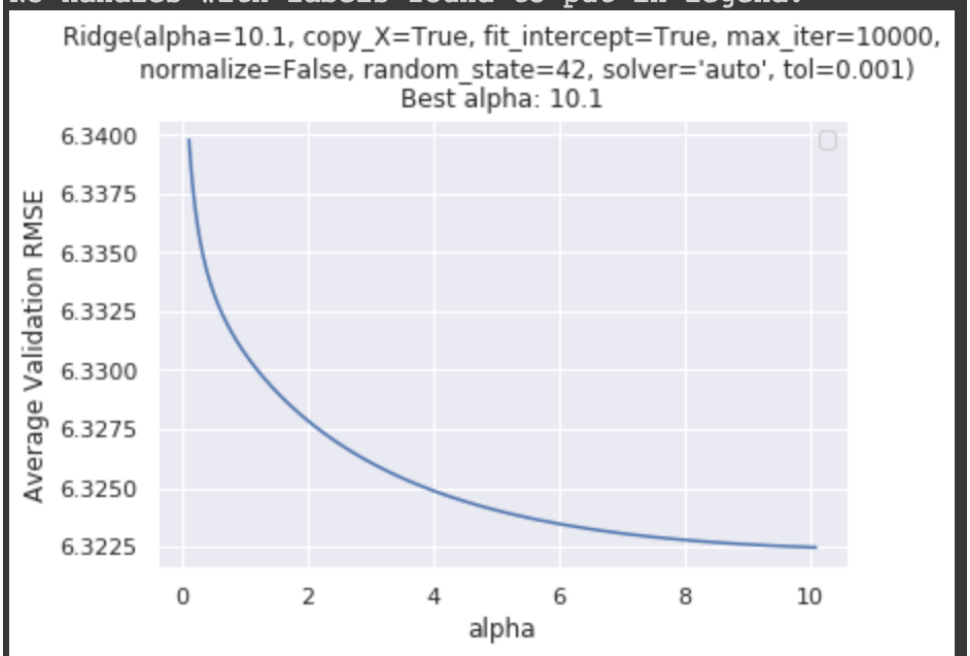
For the transcoding dataset, we've added some artificial features to the dataset. First of all, as a hyperparameter, we've determined the number of features to be added. For

every feature, we return two random numbers to determine which columns of the data will be used for nominator and denominator. The reason we've used random columns of the data is not to introduce any bias from previous experiments with the same dataset. The following code snippet helped us achieve feature crafting:

```
import random
num_features_add = 10
num_features = X_temp.shape[1]
for i in range(num_features_add):
  j = random.randint(0,num_features)
  k = random.randint(0,num_features)
  X_temp.insert(X_temp.shape[1], str(i), (X_temp.values[:,j]/X_temp.values[:,k]).tolist(), True)
  X_test.insert(X_test.shape[1], str(i), (X_test.values[:,j]/X_test.values[:,k]).tolist(), True)
```

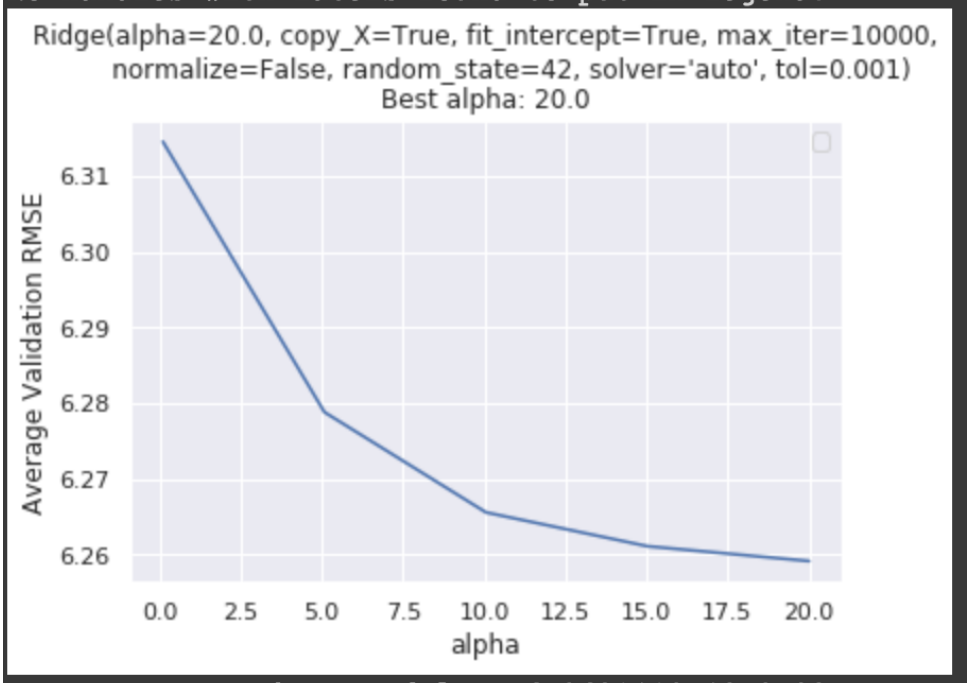
This approach would make sense as it would introduce relationships between features that otherwise would not be articulated with any of the models that we have used. From our experiments, we have seen a very slight increase in the accuracy of our polynomial regression for the transcoding-time dataset. However, due to the probabilistic nature of combined feature, it took many tries to find the so-called artificial-features. Here is the example when we added 3 more features:

```
(55027, 24)
(55027, 325)
No handles with labels found to put in legend.
```




```
Test set RMSE (best model):  7.103000804909676
```

```
(55027, 27)
(55027, 406)
No handles with labels found to put in legend.
```




```
Test set RMSE (best model):  6.368141074870708
```

From the above plots, we can see that when we add 3 more features, we have an improvement in both average validation RMSE and the test set RMSE. From the first two line of the code output, we can see that the number of features have increased from 24 to 27 and number of polynomial features have increased from 325 to 406. Please note that, when we were doing the case with more features we changed the span of the alpha values tried. So, in general, adding some features helped us decreasing the RMSE on both validation and test set.

Neural Network

A neural network is a multilayer perceptron. Artificial neural networks have two main hyperparameters that control the architecture or topology of the network: the number of layers and the number of nodes in each hidden layer.

Question 16: Neural network and linear regression

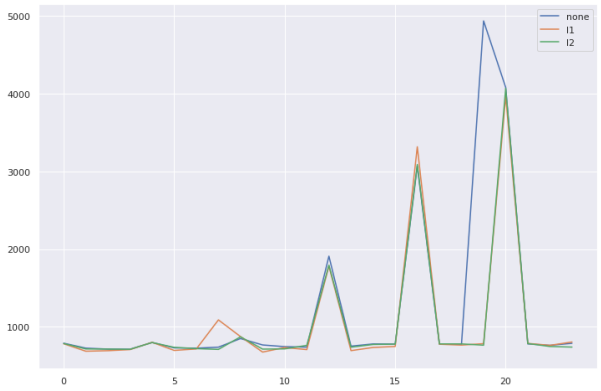
Neural networks are the very complex "evolution" of linear regression designed to be able to model complex structures in the data. Instead of a lightly parametrised function in linear regression case, neural network exploits a highly parametrised function very flexible that will be shaped conveniently during the learning phase. While there is no non-linearity in linear regression models, we could introduce a non-linearity through activation function and then hidden layers indicating new intermediate features for higher underlying complexity between inputs and outputs.

Question 17: Performance of hyper-parameter sets

Here we use linear activation function for the output layer $a = c * x$ which meaning that the inputs are multiplied by the weights for each neuron, and use relu (rectified linear activation unit) for the underlying layers. For bike sharing dataset, we first plot the RMSE vs the workflow of networks (indexed with intergers from 0 to 23), each corresponding to the different sets of hyperparameters along with different regularization. Specifically, We set the number of neurons as 512, 256, 128, 64, 32 16 and the number of layers or depth as 0, 2, 4, 6. Here is an example of the summary of the neural network with one layer.

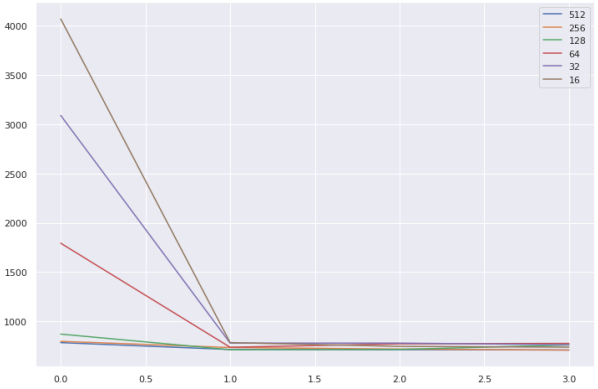
```
Parameters:  [0, 512, 'none']
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 512)               6144
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 513
=================================================================
Total params: 6,657
Trainable params: 6,657
Non-trainable params: 0
_________________________________________________________________
Batch RMSE:  1182.0363
Batch RMSE:  919.28894
Batch RMSE:  722.9854
Batch RMSE:  853.6271
Batch RMSE:  757.2072
Batch RMSE:  772.9804
Batch RMSE:  811.8798
Batch RMSE:  920.04266
Batch RMSE:  804.5649
Batch RMSE:  973.2834

Average RMSE:  871.789604836417
RMSE - Test Set:  1081.1390616256115
```

We analyze the performance using RMSE.

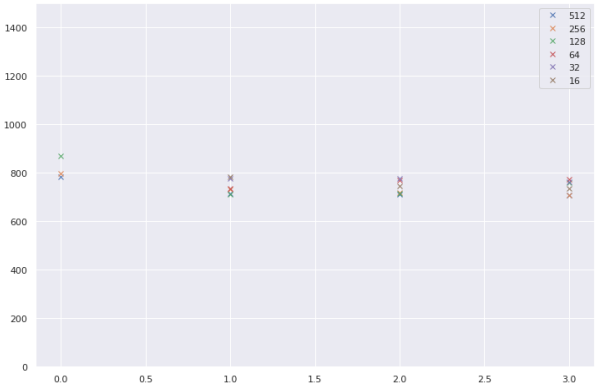


From the plot we could see that L2 regularization could improve the performance

of the network. The peaks are due to different sets of hyperparameters of the network and the regularizer, tha last one, for example, stands for the error of the most simple architecture with only 16 neurons and no hidden layer.
We further analyze the effects of the network size with l2 regularization.



The plot indicates that large number of neurons could improve the performance remarkably for small depth but not necessarily for large depth. Then the networks with RMSE below 1400 are reserved for detailed analysis and presented as discrete points below.



The figure indicates that the increase of either the depth or the number of hidden neurons does not result in better performance certainly. With more than two hidden layers in this case, the number of neurons has not explicit relation with RMSE.

Question 18: Activation function

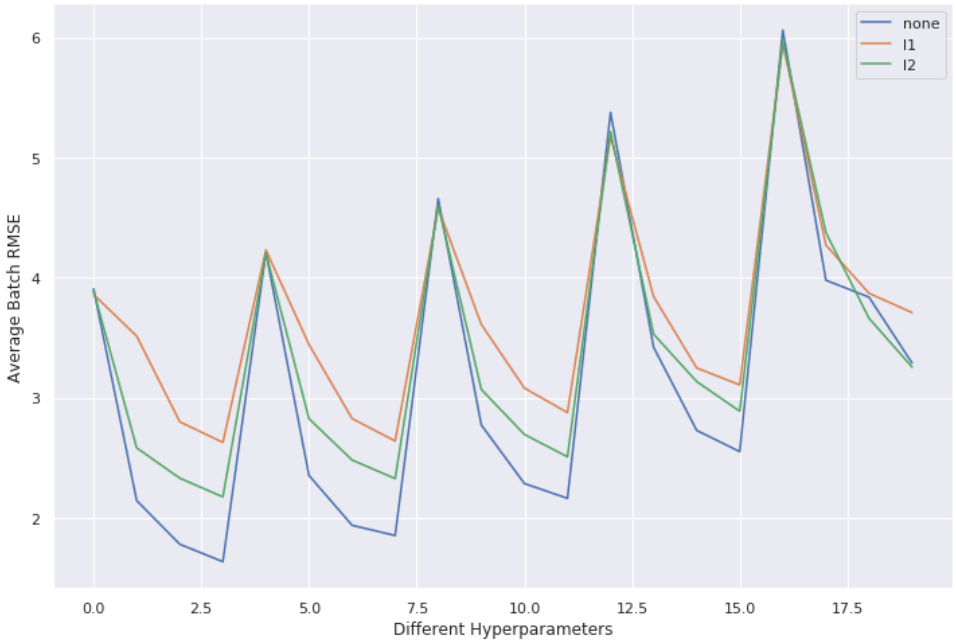
Here we use the linear activation function for the output layer. The reason we use a linear activation is not to limit the output of the network. Empirically, we've seen that other activation functions are not suitable for our purposes. For the hidden layers, we have used 'relu' activation function to introduce non-linearities to the network. The sigmoid and hyperbolic tangent activation functions might not be used in networks with many layers due to the possible problem of vanishing gradient problem.

Question 19: The depth of the network

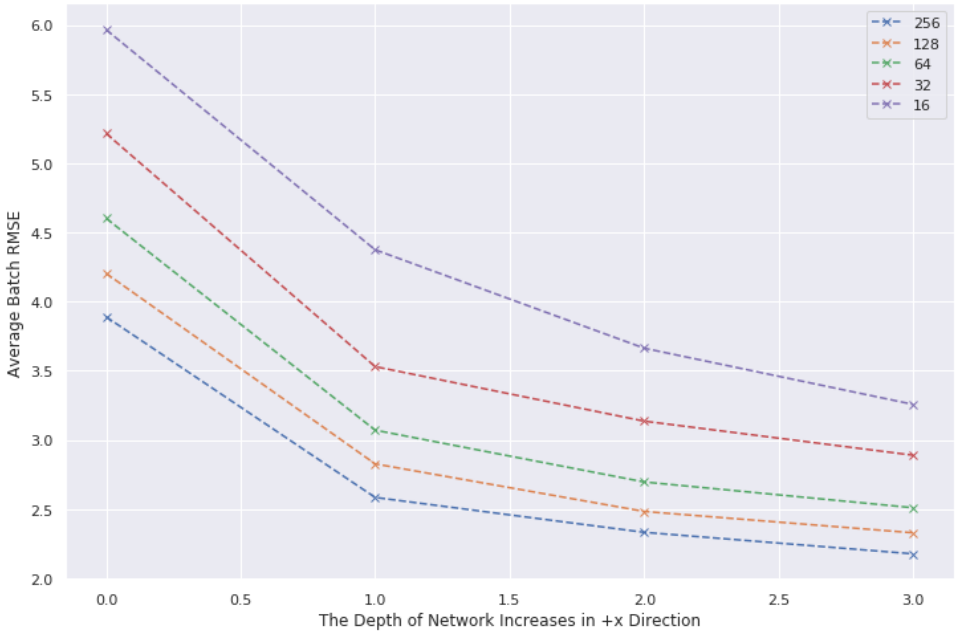
Deeper is not necessarily better and three main problems may occur as we increase the depth of the network too much. Firstly, very deep models with many parameters could cause overfitting very easily and thus perform poorly on the test data. Secondly, in very deep networks, the gradient may be vanishingly small, effectively preventing the weight from changing its value. In the worst case, this may completely stop the neural network from further training. Thirdly, it is challenging and costly in terms of compute and storage. To show that, we extract the parameter summary of the network with 6 hidden layers. In this case, the total number of paramters are 794,625 which is very large to compute but without significant improvement of the performance as stated above.

```
Parameters:  [3, 512, 'none']
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_9 (Dense)              (None, 512)               6144
_________________________________________________________________
dense_10 (Dense)             (None, 512)               262656
_________________________________________________________________
dense_11 (Dense)             (None, 512)               262656
_________________________________________________________________
dense_12 (Dense)             (None, 512)               262656
_________________________________________________________________
dense_13 (Dense)             (None, 1)                 513
=================================================================
Total params: 794,625
Trainable params: 794,625
Non-trainable params: 0
_________________________________________________________________
Batch RMSE:  693.96313
```

For video transcoding dataset, we first plot the RMSE vs the workflow of 20 networks (indexed with intergers from 0 to 19), each corresponding to the different sets of hyperparameters along with different regularization. Firstly, We analyze the performance using RMSE.



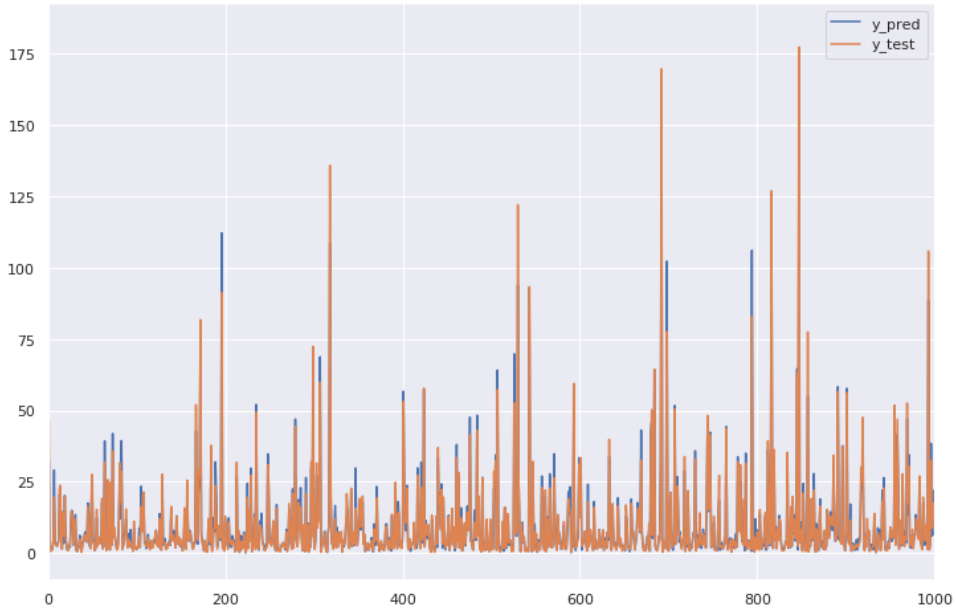
From the plot we could see that neither l1 nor l2 regularization could improve the performance of the network. Thus, we implement networks with different number hidden neurons and depth systematically without regularizer.

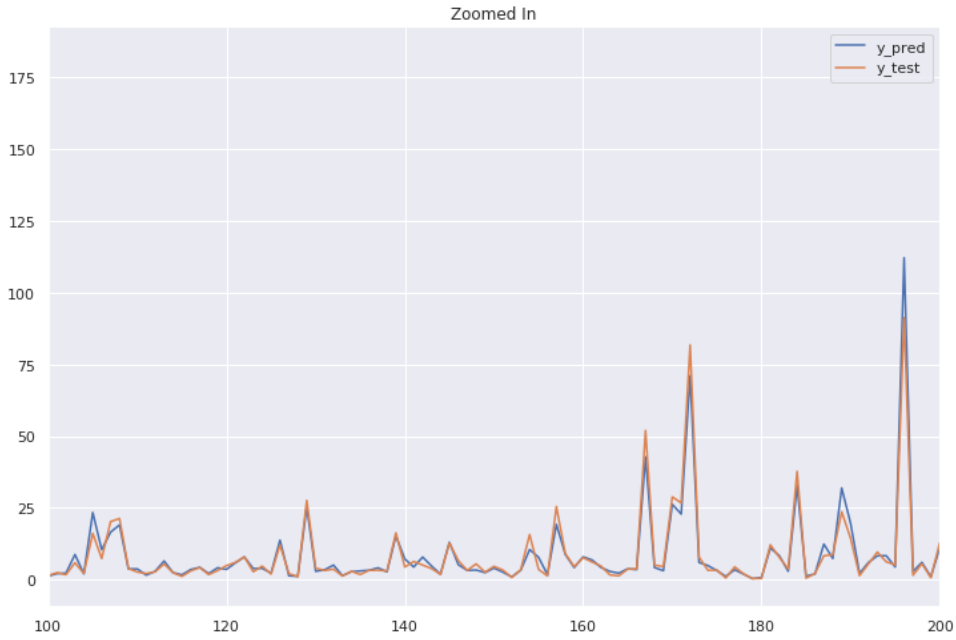


Average validation error keeps decreasing with higher depth, but shows no monotonic relationship with number of neurons. And we report the best neural network parameter summary shown as below:

```
Parameters:  [3, 256, 'none']
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_9 (Dense)              (None, 256)               6400
_________________________________________________________________
dense_10 (Dense)             (None, 256)               65792
_________________________________________________________________
dense_11 (Dense)             (None, 256)               65792
_________________________________________________________________
dense_12 (Dense)             (None, 256)               65792
_________________________________________________________________
dense_13 (Dense)             (None, 1)                 257
=================================================================
Total params: 204,033
Trainable params: 204,033
Non-trainable params: 0
_________________________________________________________________
Batch RMSE:  2.1229694
Batch RMSE:  1.7260858
Batch RMSE:  1.3770722
Batch RMSE:  1.7586702
Batch RMSE:  1.6139045
Batch RMSE:  1.5387403
Batch RMSE:  1.4510257
Batch RMSE:  1.5814012
Batch RMSE:  1.7200027
Batch RMSE:  1.4802685

Average RMSE:  1.6370141226003405
RMSE - Test Set:  1.5418589997261667
```

To visualize the predictive performance of the best model we choose, we plot the hypothesis results and true labels in the same figure as below ranging among all feature values and the zoom-in version.





Question 20:

Random forest can be thought as an ensemble method. Each decision tree in random forest acts like a model and predicts an output. The most important thing in ensemble is the correlation between predictors if the predictors correlated too much, there is no point in using multiple predictors. In other words, main intuition behind the ensemble method is since predictors are not correlated much they are prone to make mistake in different samples. Since making mistake for the same sample is rare for uncorrelated models, different predictors cover up short comings of other predictors. So that is why

ensemble can be viewed as regularization. Making decision according to multiple models decrease the chance of over fitting and helps the generalization of the data. Number of trees parameter for random forest controls the number of predictors, in other words trees as suggested by the name. As explained above random forest can be viewed as a ensemble method and the trees as the predictors.So as we increase the number of the trees we increase our regularization strength if we assume trees are uncorrelated.(which are uncorrelated in our case). So since we have multiple different trees we can say that our model is more robust to noise and better generalize the model due to the facts discussed above. So we can say that as we increase the number of trees our test rmse decreases which is exactly the same manner with the regularizes. This can be seen in the following figure:

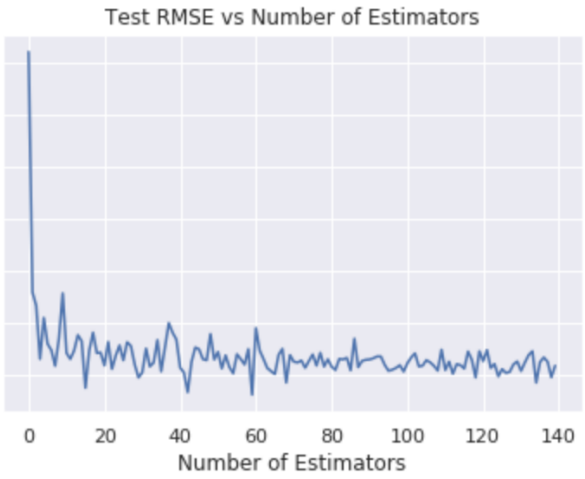


Figure 4: RMSE values for changing number of estimators while other hyper parameters are fixed

As we increase the depth of the tree, the number of splits increases and it can capture more information about the data.One should be careful while increasing max depth because large values of depth can cause overfit. So as we increase the max depth of the tree rmse decreases which can be seen in the following figure:

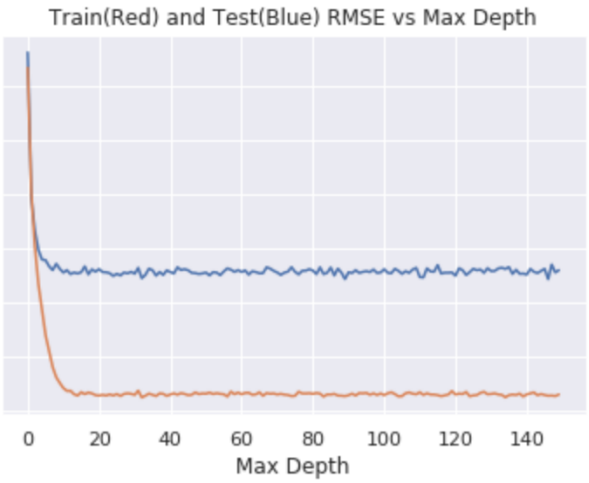


Figure 5: Test RMSE values for changing max depth while other hyper parameters are fixed

As we increase the max features of the tree, rmse value for the test decreases. This is reasonable because max feature is the number of features that is available for each node split.As we have more choice for the split we can represent the data better.This effect can be seen in the following graph:

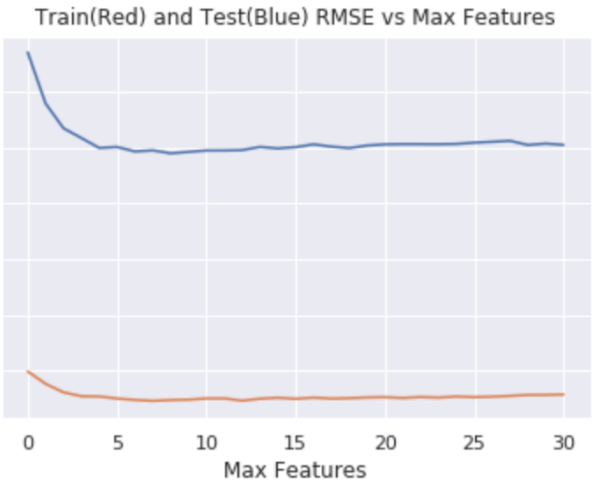


Figure 6: Test RMSE values for changing max feature while other hyper parameters are fixed

From above graph we can infer that max feature has regularization effect since as we increase it our training rmse does not change but test accuracy decreases. So this means that as we increase the max feature we make better generalization for our data set.

So we can say that max feature and number of trees have a regularization effect.

Randomforest is a computationally expensive model for that reason in order to tune our hyper-parameters, we used sklearn randomsearch algorithm with cross validation. We performed 10 fold cross validation.

Question 21: Random forest performance

Random forest has advantages such as good predictive performance, reliable feature importance estimate and efficient estimation of the test error without too much cost of training with cross-validation. Every new input is run down all of the trees and is a weighted average of all the terminal nodes, so that it could deal with unbalanced and missing data with fast runtimes.

Bike Sharing Data Results

We randomly searched from following range of values, 1-150 for max depth, 300-4000 for number of estimators, 1-32 for max features. The hyper-parameters which gives best test rmse are as follows:

```
{'max_depth': 26, 'max_features': 7, 'n_estimators': 762}
```

Our best training rmse is 230.132, validation rmse is 458.2352 and best test rmse as follows:

```
Model Performance
RMSE: 589.0865
```

OOB score for the best hyperparameters is 0.8730410679 R2 score for the best hyperparameters is 0.8842710200286167

Video Trans coding time Data Results

We randomly searched from following range of values, 1-150 for max depth, 300-4000 for number of estimators, 1-32 for max features. The hyper-parameters which gives best test rmse are as follows:

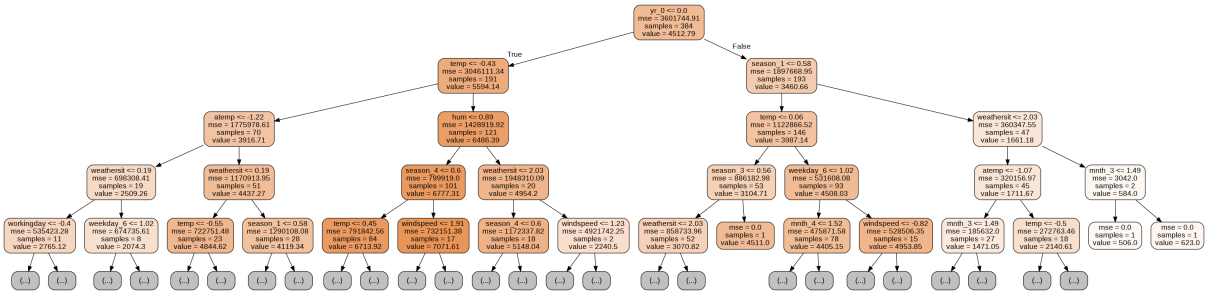
```
{'max_depth': 32, 'max_features': 8, 'n_estimators': 436}
```

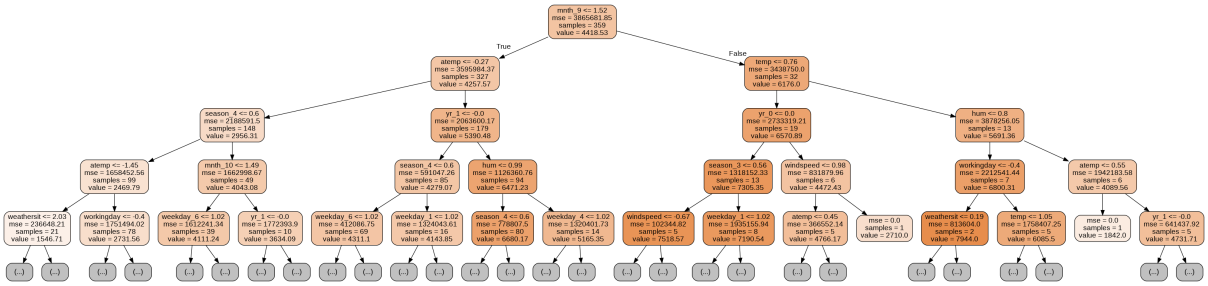
Our best training rmse is 0.5354, validation rmse is 1.24294 and best test rmse as follows:

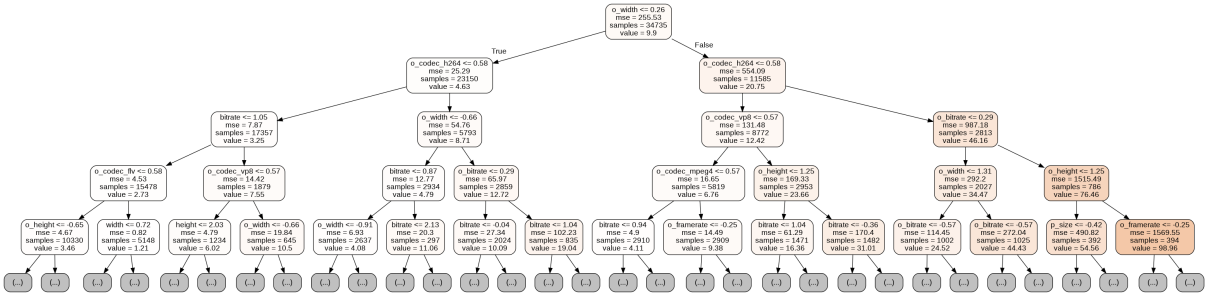
```
Model Performance
Average Error: 1.5325
```

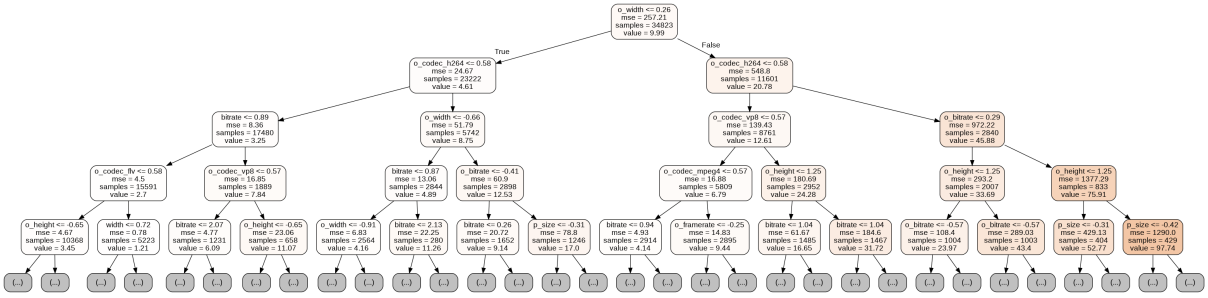
OOB score for the best hyperparameters is 0.9920276206614688 R2 score for the best hyperparameters is 0.9988845679379025

Question 22: Structure plot of random forest model









First of all, the branching changes from model to model. However, in order to address this question, let's have a look at the very first figure of this question for the bike-sharing dataset.

At the root node, we observe a selection for the year parameter. This would make sense as there would be a statistical difference in year 0 and 1. Then, in the deeper levels of the tree, we see that there are decisions made with temperature, humidity, and seasonal information. This is very intuitive as these features can really be correlated with the output. It can be inferred from the trees that if a feature is important it should appear in the upper layers.

For the transcoding time dataset, we observe a decision on the output width in the root node. Then, in the deeper levels, we observe decisions made based on the codec types, output width and height. This is also highly intuitional as the output width and height would affect the transcoding time. For the decisions made based on the codec type, it is logical to have an early seperation made based on the codec types because different codec types might have different dynamics regarding the transcoding time. In light of these observations, we can conclude that random forest regressors that we have tried with both of the datasets end up having intuitional nodes in the shallow levels of the trees. The important features that we found in the previous questions showed up in the very first layers of the trees. Another observation is that the important features might have tendency to appear in the first layers of the trees. However, this might be related to implementation as well. Here we use sklearn's implementation.

Question 23: RMSE for training and validation sets

Error on the training set let us observe how our model is progressing in terms of it's training, however error on the validation set let us get a measure of the quality of our model. In other words, how generalized is it,how well it's able to make new predictions for the data that has not seen before. So the error for training is calculated for the samples which are seen before, while error for validation is calculated with the unseen data. For that reason generally training error is lower than validation with few exceptions. So we can say that main reason for getting different error values for validation and training is training error calculated with seen data however, test error calculated with unseen data.

Question 24: OOB error

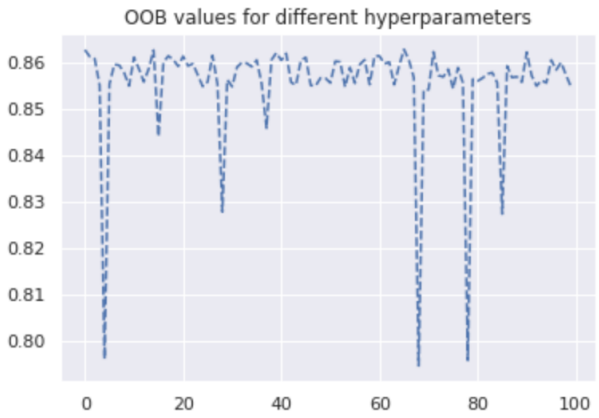
The OOB error is the average error for each data points calculated from all other trees without this point in the respective bootstrap sample. In scikit-learn, the default metric of OOB error is out of bag R2 score which is defined as $R^2(y-y_{pred}) = 1-\frac{\sum y-y_{pred}}{\sum y-y_{average}}$. It first leaves the sample to be estimated out through indices and does the error trajectory during training with R2 score of for the predictions. One can optimize the hyperparameters using OOB error because intuition behind the OOB is really similar to cross validation. Decision trees are trained with a train set and the samples which are not used to train the model are used to calculate the OOB error.One can use this error to evaluate the model and choose the best hyper parameters. Only drawback of this methodology occurs when there is not enough data because approximately 0.368 of the data is being used for validation which is too much for validation. This can be calculated as follows :

$$(\frac{N-1}{N})^N$$

For a large N it is as follows :

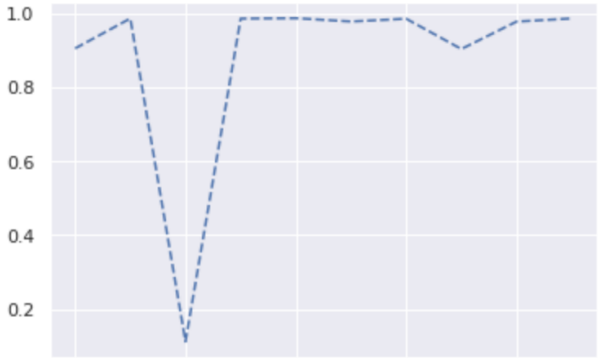
$$lim_{N-\infty}(1-\frac{1}{N})^N = e^{-1} = 0.368$$

Bike Sharing Data OOB results of randomsearch with 10 fold cross validation



OOB score for the best hyperparameters is 0.8730410679

Video Transcoding Time Data OOB results of randomsearch with 10 fold cross validation



OOB score for the best hyperparameters is 0.9920276206614688