Assignment 2 - Group: 79

'Prediction of the Burned Area in the forest'

Tutors: Iwan Budiman, Canh Dinh, Yixuan Zhang, Yang Lin, Zhiyi Wang, Shaojun Zhang, Rui Dong, Haoyu He, Fangzhou Shi, Fengxiang He, Dai Hoang Tran

Group members: Haichen Zhu (hzhu8034), Yeseul Yoon (yyoo0548), Kuo Yuan (kyua4202)

1. Abstract

This project aims to build different regression models for predicting the burned area using date, geometrical and meteorological features. Modeling focuses on how to deal with the extremely unbalanced distribution on the target field as well as a small number of instances with attributes in diverse scales. We will construct several well-known models in machine learning and compare them in terms of reasonable evaluation methods.

2. Introduction

2.1 Data Description

Forest fires data (From : https://archive.ics.uci.edu/ml/datasets/Forest+Fires) we have chosen has 12 attributes such as x/y-axis, month, day, temperature, relative humidity, and target variable 'area'. The list below shows the types of each attribute and the table on the right side is the description of our data.

```
Data Types:

X           int64
Y           int64
month      object
day        object
FFMC      float64
DMC       float64
DC        float64
ISI       float64
temp      float64
RH          int64
wind      float64
rain      float64
area      float64
dtype: object
```

	X	Y	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
count	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000
mean	4.669246	4.299807	90.644681	110.872340	547.940039	9.021663	18.889168	44.288201	4.017602	0.021663	12.847292
std	2.313778	1.229900	5.520111	64.046482	248.066192	4.559477	5.806625	16.317469	1.791653	0.295959	63.655818
min	1.000000	2.000000	18.700000	1.100000	7.900000	0.000000	2.200000	15.000000	0.400000	0.000000	0.000000
25%	3.000000	4.000000	90.200000	68.600000	437.700000	6.500000	15.500000	33.000000	2.700000	0.000000	0.000000
50%	4.000000	4.000000	91.600000	108.300000	664.200000	8.400000	19.300000	42.000000	4.000000	0.000000	0.520000
75%	7.000000	5.000000	92.900000	142.400000	713.900000	10.800000	22.800000	53.000000	4.900000	0.000000	6.570000
max	9.000000	9.000000	96.200000	291.300000	860.600000	56.100000	33.300000	100.000000	9.400000	6.400000	1090.840000

This data set has 517 data, which could be considered as small data for accurate prediction, so we might have to make our sample bigger by up-sampling. One more thing we could capture from the data description was that the maximum value of the burned area is 1090.84, although the mean value is just nearly 12.85, and the majority of data(47.776%) has zero value. Hence, we assumed that this data set would be challenging due to this biased data. Another must-do step for the data pre-processing is finding a missing value and deal with it. Fortunately, our data set doesn't seem to have any missing value and the related function could prove so.

Now, looking at the meteorological attributes contained in the data set, the temperature values are distributed as the gaussian distribution like below, and this might tell us that this feature could be one of affective on our future learning model. On the other hand, RH(Relative Humidity) shows the Gaussian distribution but the data seems more left-skewered. Moreover, RH has a much wider range of data compared to the temperature. Interestingly, we found out the rain data is highly biased to 0 and we assume that this causes very low correlation with other factors and would not be used for modeling. Lastly, the wind is relatively evenly distributed.

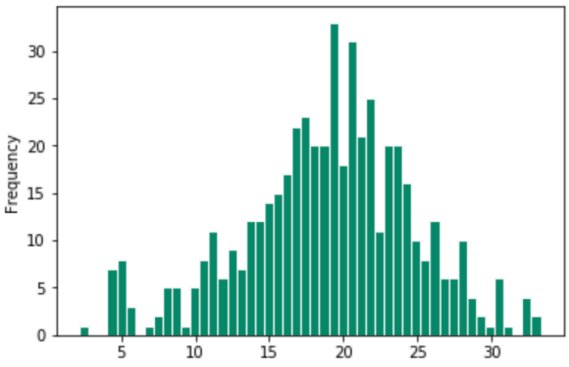


Figure 1. Histogram of Temperature (◦C)

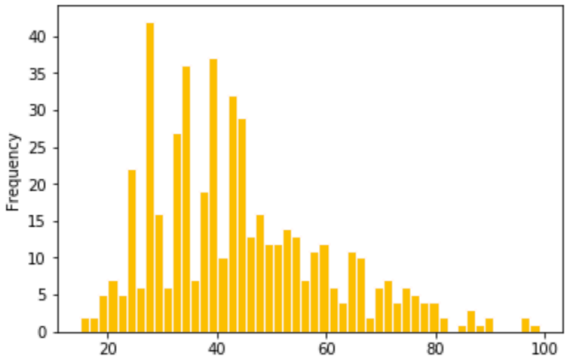


Figure 2. Histogram of Relative Humidity (%)

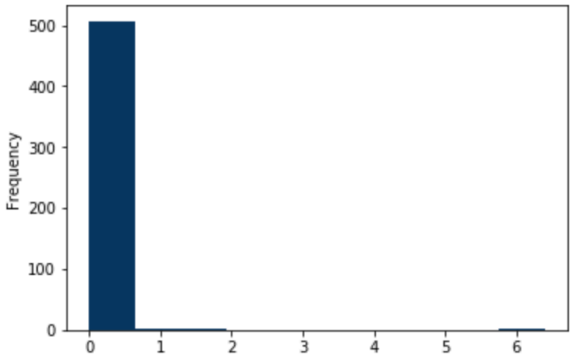


Figure 3. Histogram of Rain (mm/m2)

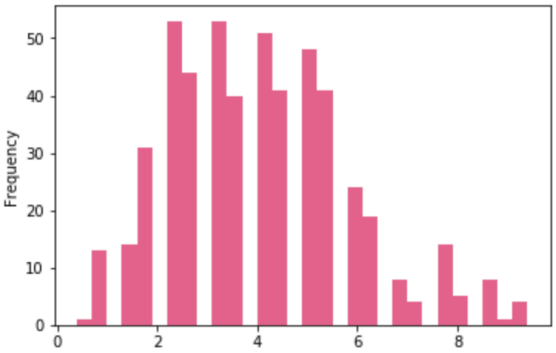


Figure 4. Histogram of Wind (km/h)

Next, let's move on to the date data. The majority of fire has happened in August and September and the frequencies of each day are fairly evenly distributed.

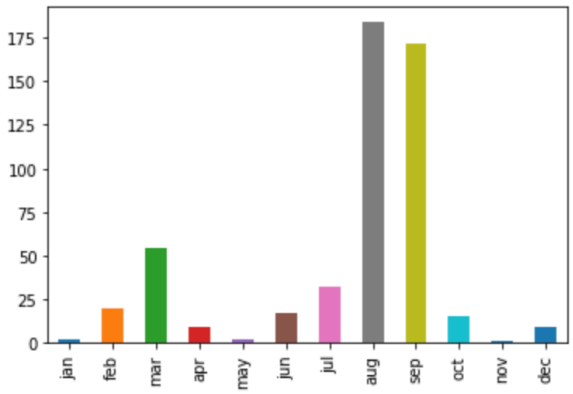


Figure 5. Histogram of month

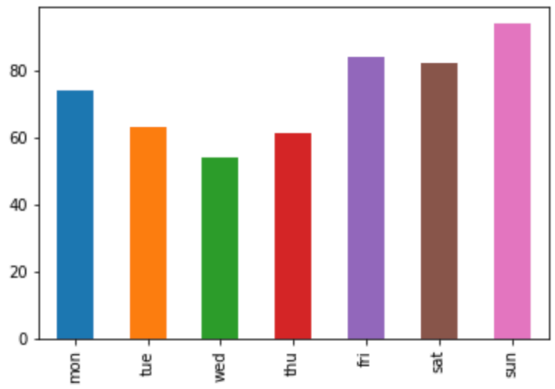


Figure 6. Histogram of day

We faced the challenging problem which is highly left-skewed burnt area data. Although we could see the target data is extremely biased ranging from 0 to over 1000 and this led to that skewed histogram graph like below.

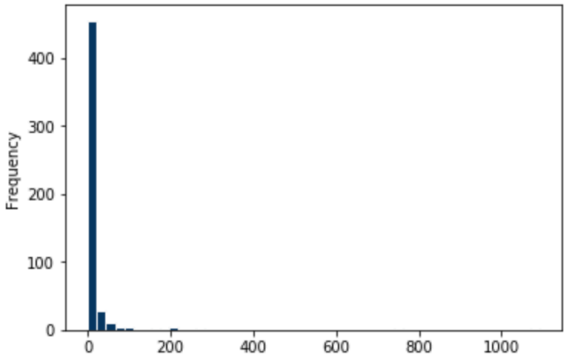


Figure 7. Histogram of the burned area

The overall density graphs below illustrate all the attributes that we already had a look using the frequency charts. These graphs could be great indicators to give us an overview of our data set.

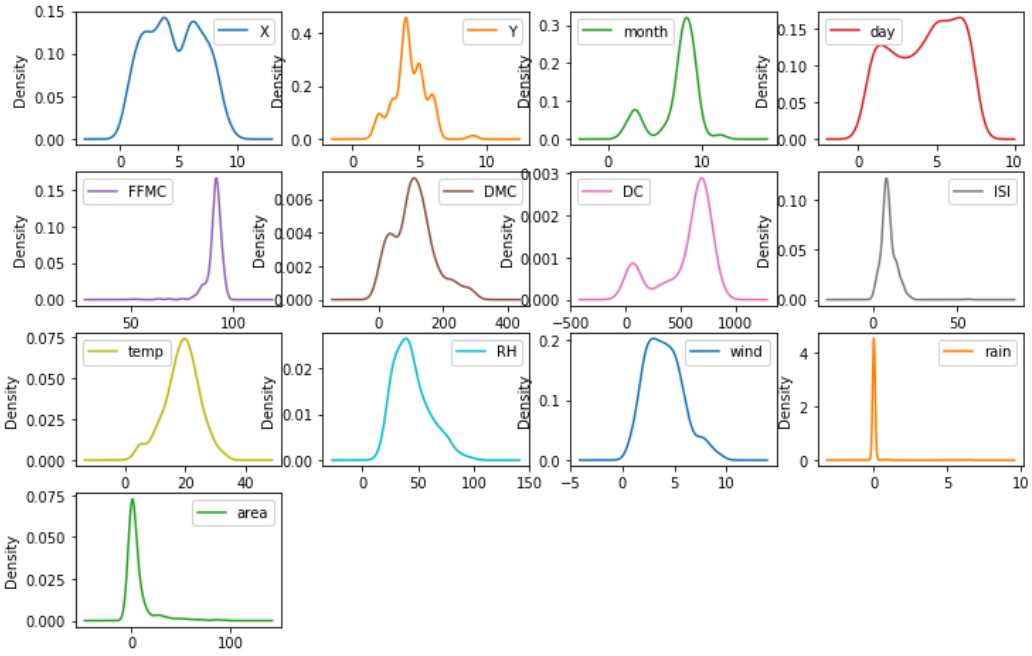


Figure 8. Density graphs for all attributes

The graph below shows the correlation between area and all the attributes in our data set. Apart from some features related to each other, any other features seem to have relatively less correlation.

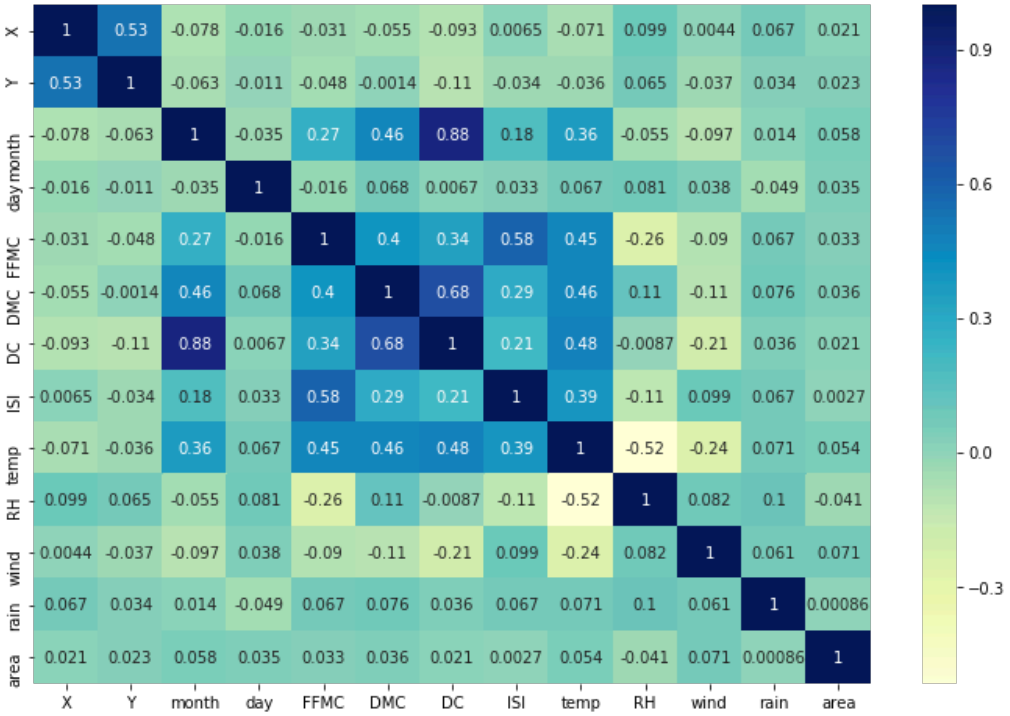


Figure 9. Correlation heat map

2.2 Classifiers

For prediction, we will implement three different classifiers, such as Decision Tree, Random Forest, and Lasso Regression.

3. Pre-Processing

3.1 Outliers

First of all, we decided to deal with the outliers which can cause a bad influence for modeling. As mentioned before, the most of incidents were in August and September, and several months have only 1-2 counts of fire. Hence, we considered the less-frequent months as outliers and removed them before we move on to the training.

```
1  data.month.value_counts()
```

```
aug    184
sep    172
mar     54
jul     32
feb     20
jun     17
oct     15
dec      9
apr      9
jan      2
may      2
nov      1
Name: month, dtype: int64
```

```
1  remove_month = ['nov','jan','may']
2  fire_data = data.drop
3  (data[data.month.isin(remove_month)].index ,axis=0)
4  fire_data.month.value_counts()
```

```
aug    184
sep    172
mar     54
jul     32
feb     20
jun     17
oct     15
dec      9
apr      9
Name: month, dtype: int64
```

From the counts result on the left hand side above, we see that January, May and November have relatively low counts for fire incidents so we removed the data belong those months.

3.2 Data Type Transformation

Another issue we have faced in this project was the data type transformation. Although some classifiers such as the decision tree detects the categorical data, not all the methods can do so. In oder to use month and day data for all of our prediction models, we converted the categorical data into the numerical data like below.

```
fire_data_num = fire_data
fire_data_num.month.replace(('jan','feb','mar','apr','may','jun','jul','aug','sep','oct','nov','dec'),
                            (1,2,3,4,5,6,7,8,9,10,11,12), inplace=True)
fire_data_num.day.replace(('mon','tue','wed','thu','fri','sat','sun'),(1,2,3,4,5,6,7), inplace=True)
```

3.3 Logarithm Function

The fire area represents a positive skew and the majority of the burned area is a small number. Regarding the data set after removing the outliers, there are 243 data with 0. To reduce skewness and improve symmetry, the logarithm function y=ln(x+1), which is a common transformation that tends to improve aggression result for skewed target[1], we applied the logarithm function to the area attribute.

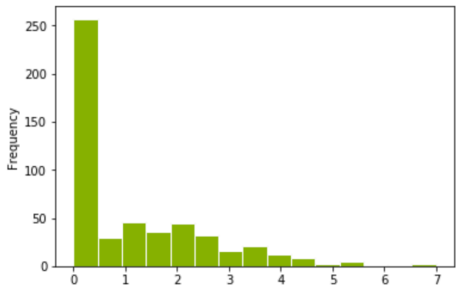


Figure 10. Histogram of new area data (logarithm)

From the logarithm function transformation, we got this less skewed frequency graph of burned area like above.

[1] Cortez P, Morais Guimar˜aes, Portugal A. A Data Mining Approach to Predict Forest Fires Paulo Cortezusing Meteorological Data. Guimaraes, Portugal;

3.4 Up-Sampling

Taking into account of the fact that the machine learning algorithms are opt to work better when the samples are evenly distributed, we implemented Random Up-Sampling. This provides us more balanced dataset and along with avoiding the overfitting issue. The brief description of its steps - 1. According to each algorithm, set target attribute in groups. 2. Exploring how unbalanced they are and decide how many samples for each group should be used during the training process. By experiments, models have different performance when they used the up-sampled set. The detail for each of them will be mentioned in the following part.

4. Decision Tree

4.1 Description

A decision tree is a decision support tool and easily interpreted by humans, as long as the tree is not too many branches. In terms of the size of our data set, which is relatively small, we thought we could fit our prediction model using a decision tree. Another advantage of this model is that it can deal with both numerical and categorical features. Since the forest fire data set has both variables, we assumed that this method could work well and effectively on our data set and decided to use this classifier. Moreover, a decision tree is a non-parametric method, meaning it has no assumptions about space distribution and the classifier structure. Based on this tree model, we assumed that we would be able to easily implement ensemble algorithms like AdaBoost regressor.
When we face the regression task using a decision tree could be different from the classification tasks. However, there are two same core questions to be addressed: 1. How to choose a partition point? 2. How to determine the output value of the leaf node?

A tree algorithm corresponds to a division of the input space (features space) and the output value on the division unit. In the classification tree, we use the method in information theory to select the best dividing point by calculating information gain, which can be defined as below: [2]

$$I(X;Y) = H[Y] - H[Y|X]$$

$$H[Y] = -\sum_{i=1}^{k} Pr(Y = a_i) log Pr(Y = a_i)$$

$$H[Y|X] = \sum_{k} P(X = a_k) H[Y|X = a_k]$$

In the regression tree, a heuristic approach is used (Squared loss minimisation). If we have n features, each feature has values, then we traverse all the features, try all the valuesof the feature, on the space The division is performed until the value s of the feature j is taken, so that the loss function is minimised, and thus a division point is obtained. The formula describing the process is as follows:

$$min[minLoss(y_i, c_1) + minLoss(y_i, c_2)$$

$$c_m = ave(y_i | x_i \in R_m)$$

Loss Function is defined as squared loss function:

$$Loss(y, f(x)) = (f(x) - y)^2$$

[2] Nguyen HT. [Unpublished lecture notes on Machine Learning and Data Mining]. University of Sydney; notes provided at lecture 2019 Oct 2.

4.2 Previous Work

We have found a similar work called 'Predict The Burned Area Of Forest Fires'[3] on Kaggle. In this notebook, the author simply tried several regressors in sklearn without splitting the train and test set. In this situation, the performance of decision tree regressor (CART) is perfect, achieving nearly 1 in explained variance score and less than 0.1 of mean absolute error. However, we assumed that this is due to overfitting, and it might result in low performance on new data. In light of this, we will split the data set in the beginning to make our model can not touch the test set, and deliver some feature engineering to make a simpler tree to avoid overfitting.

4.3 Pre-processing

While too many numbers of features can result in overfitting in a small data set, it can be sensible to do Recursive Feature Elimination (RFE), prior to training our model. RFE is a method recursively remove features and work on the remain data to build a model[4]. In this step, we removed relatively irrelevant features 'Y', 'month', 'day', 'FFMC' and 'rain'.

4.4 Experiment

To get a better prediction decision tree regressor, the right ratio of samples to a number of features is important, which can be modified through parameter max_features in the DecisionTreeRegrssor() method.

In order to gain an insight of the tree structure and how the decision tree makes predictions, which is also important for understanding the important features in the data, we initially assigned the max_depth parameter to 4 and used the export function to visualise our tree model. This number could be changed later on to obtain a better tree model. Moreover, it is notable that it doubles the cost for each additional level to populate the tree, so we used this parameter in order to control the size of the tree and also prevent over-fit.

The two useful parameters, 'min_samples_split' and 'min_samples_leaf', were considered to ensure that multiple samples guide every decision in our tree model. In other words, the smaller number in these two parameters can result in over-fitting, whereas the bigger number can prevent our model to learn from the data[5]. The 'min_samples_split' can produce small-sized leaves, while another parameter (min_samples_leaf) will make sure that the minimal size of each leaf, so that it can prevent over-fitting and low-variance problem in regression. In our experiments, these two parameters were initially assigned with 5 and 4 respectively according to the size of our data set.

Due to the imbalance of our data set, parameter 'min_weight_fraction_leaf' is possible to help to make the tree predictions less biased towards the dominant class (area = 0 in our case). The range of this parameter is [0, 0.5], which will be tested in the fine-tuning process. In light of the importance of these parameters, we need carefully determine each value. There is a GridSearchCV() class under sklearn.model_selection, which will traverse all the possible combinations of parameter values and return the best solution with cross validation. The chart below is our parameters value selection with 10-fold cross-validation. The performance seems acceptable with about 70 root mean squared error (RMSE), and the mean absolute deviation (MAD)

---

[3] Ahiale D. Predict The Burned Area Of Forest Fires. [Kaggle online notes] 2017 Dec 8

[4] Abeel T, Helleputte T, Van de Peer Y, Dupont P, Saeys Y. Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. Bioinformatics. 2009 Nov 25;26(3):392-8.

[5] Kotsiantis SB. Decision trees: a recent overview. Artificial Intelligence Review. 2013 Apr 1;39(4):261-83.

means the predicted area covers a certain range. We plot a scatter graph to compare with true values and our predicted values, and draw a line to demonstrate the regression error characteristic (REC)[6] curve.

Model		Previous Peer Work	Our Model			
			model 1	model 2	model 3	model 4
Modeling	Cross Validation	-	√	√	√	√
	Log Transformation	-	-	√	-	√
	Feature Selection	-	√	√	√	√
	Up-sampling	-	-	-	√	√
	Function	DecisionTreeRegressor()				
	Tuned Parameters	-	max_depth': 3, 'max_features': 4, 'min_samples_leaf': 3, 'min_samples_split': 2,	max_depth': 3, 'max_features': 5, 'min_samples_leaf': 2, 'min_samples_split': 2,	'max_depth': 7, 'max_features': 6, 'min_samples_leaf': 5, 'min_samples_split': 2,	'max_depth': 9, 'max_features': 3, 'min_samples_leaf': 2, 'min_samples_split': 2,
Model Performance	MSE		3973.496	8318.045	9557.379	5119.051
	RMSE		63.036	91.203	97.762	71.548
	MAD		0.000	0.000	23.617	11.886
	MAE	0.096	18.824	15.638	28.00205938	20.050
	NLL		9.69623E+34	3.25E+36	189.564	535.897
	R ^ 2		-0.001	-0.022	-0.188	-0.145
	AUC		0.293	0.712	0.453	0.568
	Running Time (s)		0.061	0.056	0.056	0.053
Link		https://www.kaggle.com/elikplim/predict-the-burned-area-of-forest-fires	-	-	-	-

Table 1. Model comparison - Decision Tree

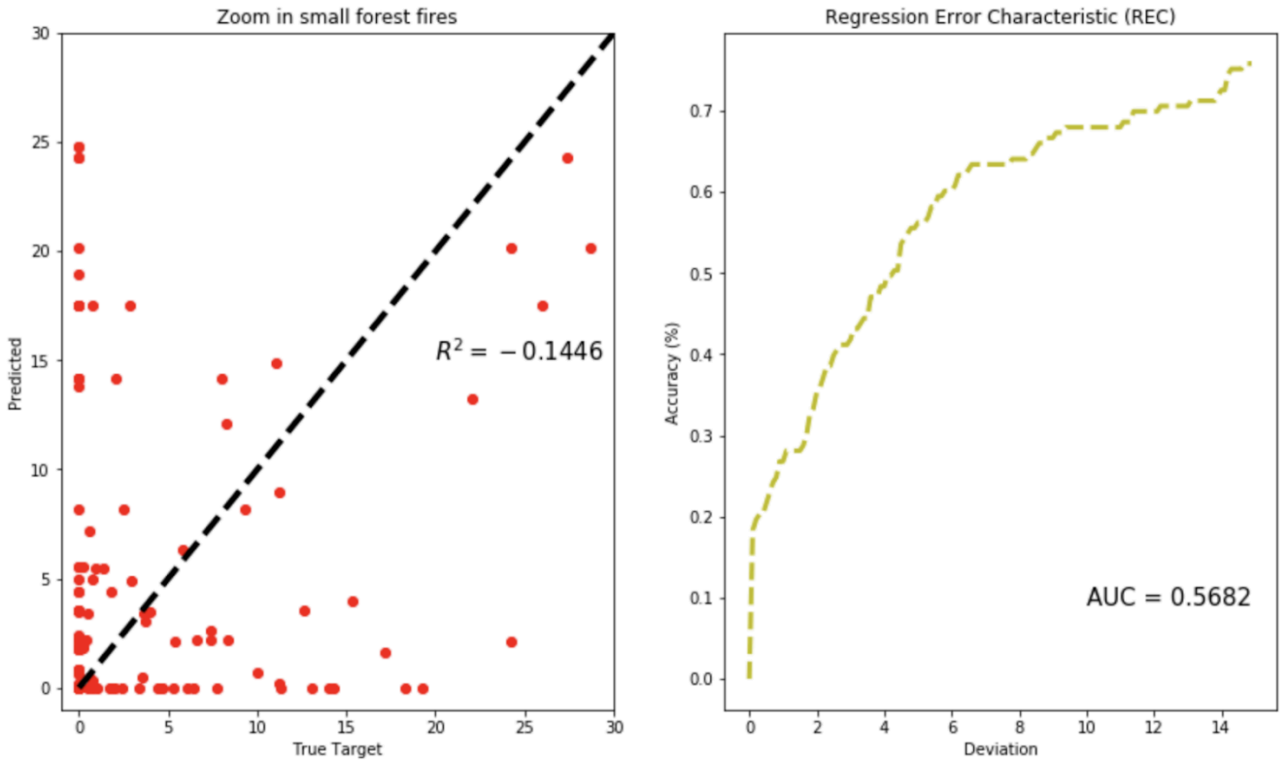


Figure 11. Model Evaluation - Decision Tree

[6] Bi J, Bennett KP. Regression error characteristic curves. InProceedings of the 20th international conference on machine learning (ICML-03) 2003 (pp. 43-50).

5. Random Forest

5.1 Description

A Random Forest is a meta-estimator using a number of results from multiple sub-samples of the data sets. This classifier constructs several decision trees while it is being trained and averages each prediction from an individual decision tree as an output. As an aspect of combining decision trees in this algorithm, Random Forest is also regarded as an ensemble model. Moreover, the trees sometimes tend to be constructed very deep with irregular patterns and result in overfitting. Here, Random Forest corrects this overfitting problem doing aggregating or averaging. By training different part of the same training set, it could reduce the variance. Furthermore, this is relatively effective when the users face the data containing a large portion of missing data and it even maintains the accuracy at the same time.

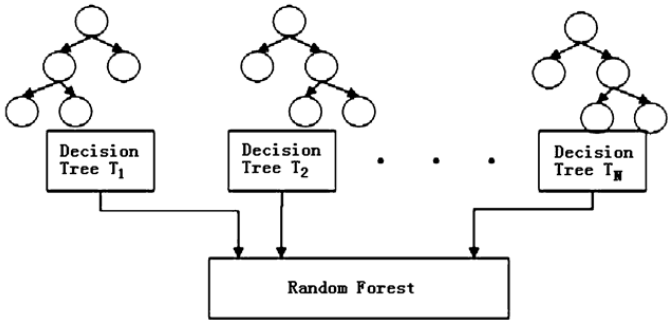


Figure 12. Random Forest [7]

Random forest method can interpret numerical, categorical, discrete, and continuous variables and it automatically implements interaction among them. Another remarkable feature of Random Forest is that trees here are insensitive to unevenly distributed data (just like what we have for this project), missing values, and also outliers. Hence, we chose Random Forest as one of our classifiers.

5.2 Previous Work

From a prediction work[8] using Random Forest method, the author certainly addressed the author certainly addressed the model was overfitted. One work resulted in 9.067032 in mean absolute error but our model outperformed as 7.1725. MAE(Mean Absolute Error) denotes an average of 'absolute' differences among the target and the predicted result and a linear score considering all the individual differences as an equally-weighted. Mean square error, which is also generally used, tends to more sensitive to outliers than MAE. This indicates how important to split the training and the test set, in terms of the way to split the data set between the previous work and our work.

5.3 Pre-Processing

Even though Random Forest method improves the overfitting issue compared to how the decision tree performs as mentioned above, it still might result in the overfitting on noisy or skewed data. While we have heavily skewed target data (area), we used stratified sampling as implemented in StratifiedShuffleSplit cross-validator for model selection in order to ensure that relative class frequencies are approximately preserved in each train and validation fold. [9]

---

[7] Mohtadi BF. In depth parameter tuning for random forest. Published on All thing in AI. 2017 Dec 22.

[8] Ahiale D. Predict The Burned Area Of Forest Fires. [Kaggle online notes] 2017 Dec 8

[9] 3. 3.1. Cross-validation: evaluating estimator performance — scikit-learn 0.21.3 documentation [Internet]. Scikit-learn.org. [cited 1 November 2019]. Available from: https://scikit-learn.org/stable/modules/cross_validation.html

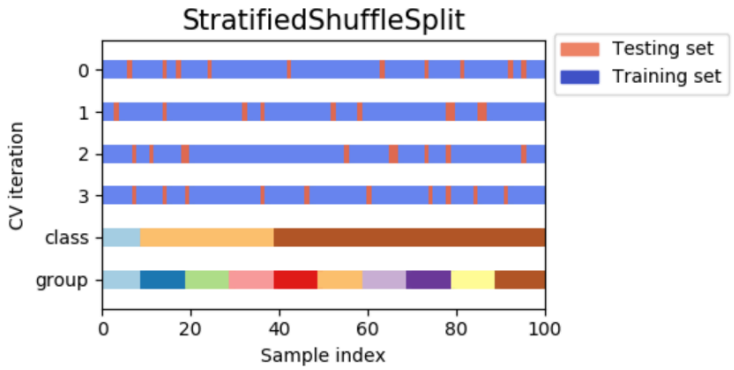


Figure 14. Stratified Shuffle Split Explanation[10]

Looking at the image below, stratified sampling returns splits by maintaining the same proportion for each target class. Specifically, we used the 'area_bins' attribute as an indicator which we already created beforehand in order to remove the outliers.

5.4 Experiment

Here, we fit the models using split training set to find out two best hyper parameters, 'max_depth' and 'n_estimators'. They respectively mean the maximum levels in each decision tree and the number of trees in the forest. Ideally, the model has more trees may perform better, however, adding too many trees could slow down the execution time for training. Moreover, the deeper the tree, the more specific information captured from the data. This model perfectly fits all the training data, but it could result in overfitting and also fail to be generalised for the test data, on the other hand. We tested 3 to 6 max_depth values and 10, 50, 100, 500, 1000 n_estimators values using GridSearchCV with 10 cross validation and fit the model again with the found best parameters for further prediction process.

As mentioned in previous parts, we made our samples larger by up-sampling, and we experimented in two different ways - original training and test set and up-sampled training and test set so as to see if the up-sampling methods help us.

Model		**Previous Work**	**Our model**	
		Peer	**Model 1**	**Model 2**
Modeling	Cross Validation		k=10	k=10
	Feature Selection		O	O
	Up-Sampling		–	O
	Function	**Random Forest**		
	Optimal n_estimator	–	1000	50
	Optimal max_depth	–	3	6
Model Performance	MSE	–	298.9015061	54.82153935
	RMSE	–	17.28876821	7.404156897
	MAD	–	0.260096829	1.723959428
	MAE	9.067032	7.172506192	5.114367137
	NLL	–	70729.53298	450.966492
	R^2	–	-0.1173	0.0051
	AUC	–	0.7355	0.6813
	Running Time	–	5.5mins	4.81 mins
Link		https://www.kaggle.com/elikplim/predict-the-burned-area-of-forest-fires	–	–

Table 2. Model comparison - Random Forest

[10] [Internet]. [cited 2 November 2019]. Available from: https://scikit-learn.org/stable/modules/cross_validation.html

Looking at the chart above, our model obviously outperformed in terms of MAE compared to the previous work. We anticipated the up-sampling could give a positive impact on the performance, and the model2 generally shows better performance from some aspects. For example, this model significantly improved error rates such as MSE, MAD, and MAE. Whereas, it was counterproductive in terms of AUC meaning that this model is less capable in distinguishing. The graphs below illustrate how the model1 and model2 worked on our data set.

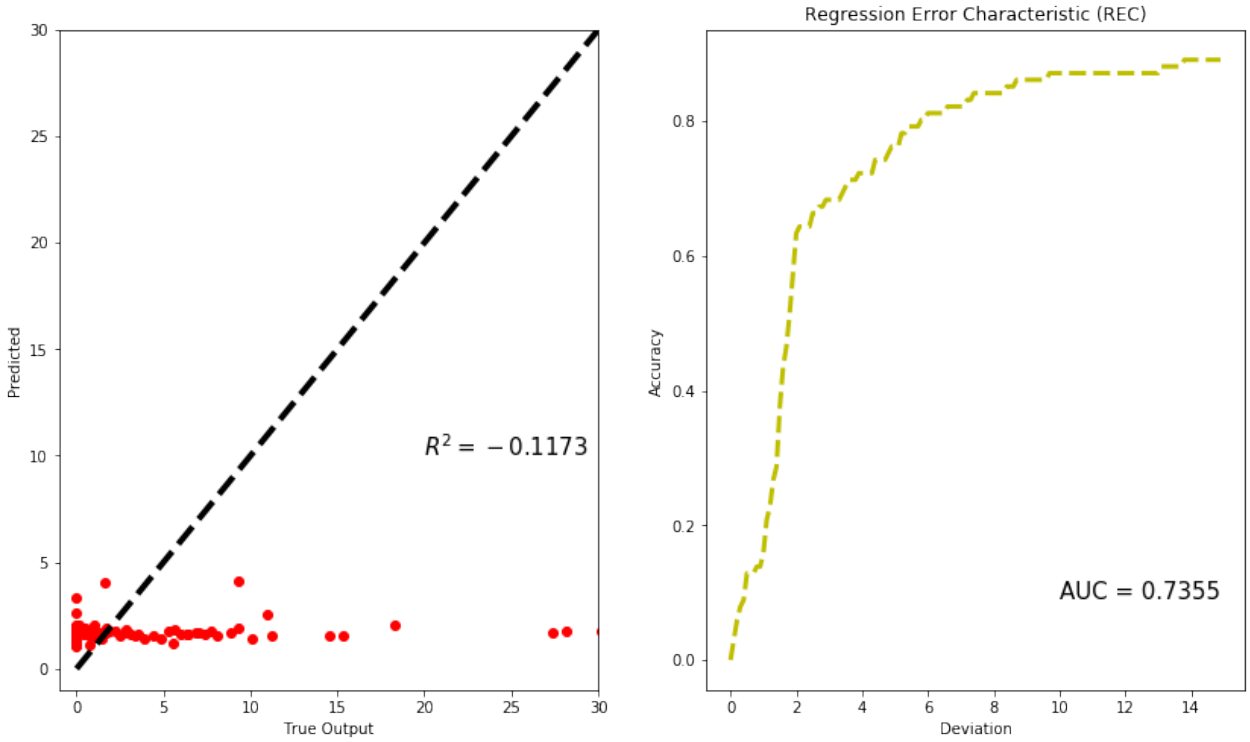


Figure 15, 16. Model Evaluation - Random Forest on original data set

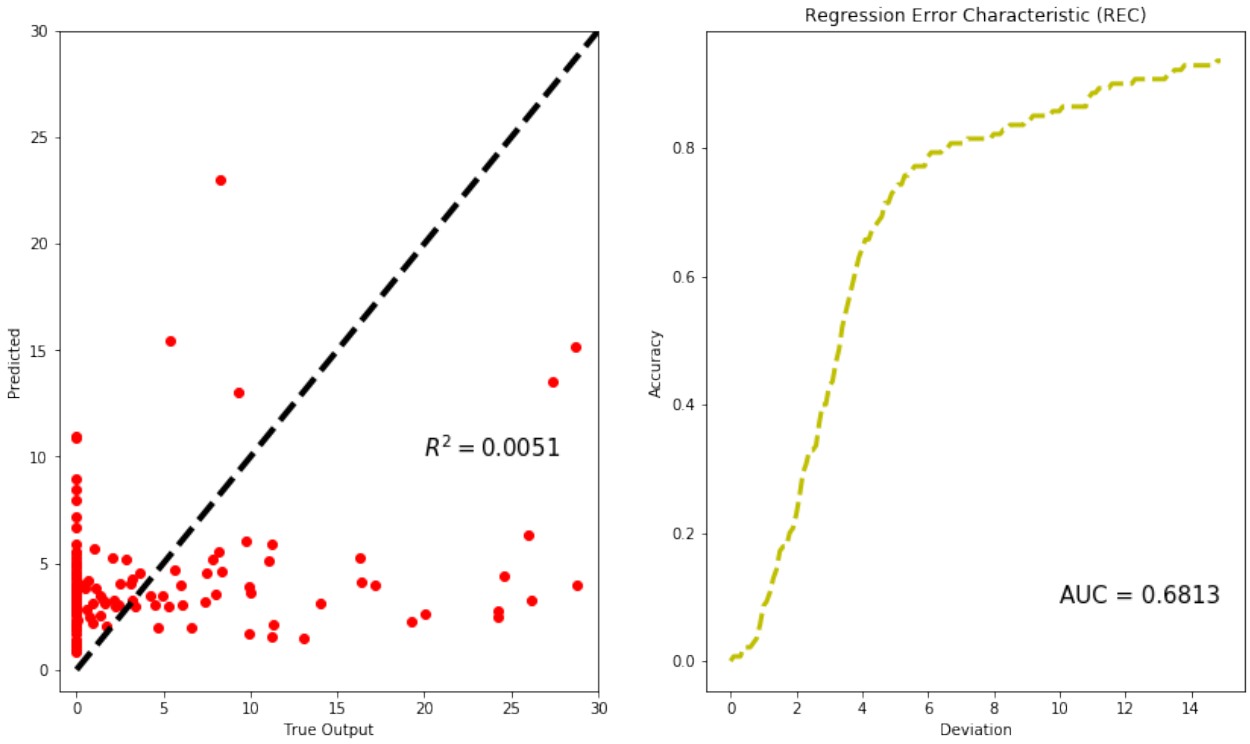


Figure 17, 18. Model Evaluation - Random Forest on up-sampled data set

6. Lasso Regression

6.1 Introduction

Lasso regression is a type of linear regression that uses shrinkage which is where data values are shrunk towards a central point, like the mean. The lasso procedure encourages simple, sparse models[11](i.e. models with fewer parameters). This particular type of regression is well-suited for models showing high levels of multicollinearity or the plan for automating certain parts of model selection, like variable selection/parameter elimination.
Compared with L2 regularisation (Ridge Regression), Lasso regression, as L1 regularisation, adds a penalty equal to the absolute value of the magnitude of coefficients. This type of regularisation can result in sparse models with few coefficients;
The goal of the algorithm is to minimise $\sum_{i=1}^{n}(y_i - \sum_{j} x_{ij}\beta_j)^2 - \lambda \sum_{j=1}^{p} |\beta_j|$, Which is the same as minimising the sum of squares with constraint Σ |Bj≤ s. Some of the βs are shrunk to exactly zero, resulting in a regression model that's easier to interpret. A tuning parameter, λ controls the strength of the L1 penalty. λ is basically the amount of shrinkage:

- When λ = 0, no parameters are eliminated. The estimate is equal to the one found with linear regression.
- As λ increases, more and more coefficients are set to zero and eliminated (theoretically, when λ = ∞, all coefficients are eliminated).
- As λ increases, bias increases.
- As λ decreases, variance increases.

6.2 Previous Work and our Experiments

We have found two highest rank previous peer works on Kaggle.
RankNo.1LinearR: for representing the linear regression model in Rank No.1[12]
RankNo.2LinearR: for representing the linear regression model in Rank No.2[13]
The following part focuses on evaluating peer works with our models.

For peer works, RankNo.1LinearR and RankNo.2LinearR, both of them did not apply cross-validation and SMOTE (up-sampling and down-sampling). RankNo.1LinearR encoded 'month' and 'day' data. At the same time, it applied feature selection and uses top 3 features in rank in the final model. The lasso linear regression model achieves 19.054901 on mean absolute error. Compared with RankNo.1LinearR, RankNo. 2LinearR focused on dealing with data attributes on many different scales and removing outliers on the target field. Through normalisation and setting new features on grouping 'area' achieved 13.408 on the root mean absolute error.
Linear regression models in this project are trained by different baselines and compared each of with others and peer previous works. As shown in the table above, there are five models based on different model tuning skills and model performances are significantly different from some others.
Model 1 based on preprocessing dataset without doing scaling, feature selection, and up-sampling and down-sampling. Compared with the other four models and peer worlds, Model 1 achiever the best RMSE(8.244) and the low MAE(5.319) and also low MAD(2.17). As shown below, the left plot shows that the regressor could make an efficient prediction when the 'area' tends to get a relatively bigger fire area.

---

[11] Pokrass J, Bronstein AM, Bronstein MM, Sprechmann P, Sapiro G. Sparse modeling of intrinsic correspondences. InComputer Graphics Forum 2013 May (Vol. 32, No. 2pt4, pp. 459-468). Oxford, UK: Blackwell Publishing Ltd.

[12] Ahiale D. Predict The Burned Area Of Forest Fires. [Kaggle online notes] 2017 Dec 8

[13] TravelCodeSleep. End-to-End Regression Pipeline Using ScikitLearn. [Kaggle online notes] 2019 Jan

Model		Previous Peer Works		Our Model				
		Peer 1	Peer 2	Model 1	Model 2	Model 3	Model 4	Model 5
Modeling	Cross Validation	–	–	**k=10**	k=10	k=10	k=10	k=10
	Normalisation	–	●	–	–	●	●	–
	Feature Selection	●	–	–	●	–	●	–
	Up-Sampling	–	–	–	–	–	–	●
	Function	Lasso()	LinearRegression()	linear_model.LassoCV()				
Model Performance	MSE	–	–	**67.971**	69.671	68.17	70.638	16896.094
	RMSE	–	13.408	**8.244**	8.347	8.257	8.405	129.985
	MAD	–	–	**2.17**	1.754	0.463	0.27	49.382
	MAE	19.054901	–	**5.319**	5.617	5.056	5.137	68.083
	NLL	–	–	**592.709**	952.609	12889.734	48438.644	189.321
	R^2	–	–	**0.0268**	0.0025	0.024	-0.0114	-0.4764
	AUC	–	–	**0.6857**	0.6668	0.711	0.709	0.1136
	Running Time	–	–	**1.1524152**	1.0782811	1.1341686	1.1187691	1.3831279
Link		https://www.kaggle.com/elikplim/predict-the-burned-area-of-forest-fires	https://www.kaggle.com/elikplim/forest-fires-data-set/kernels	–	–	–	–	–

Table 3. Model comparison - Lasso Regression

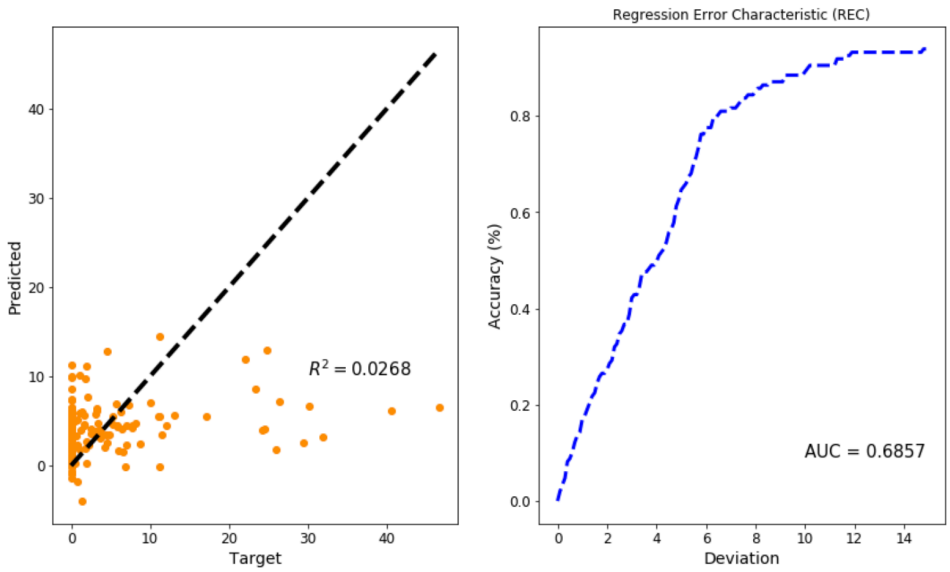


Figure 19, 20. Model Evaluation - Lasso Regression-1

Next, Model 2 based on preprocessing dataset with feature selection only. Compared with Model 1, it selects the most positive correlated features for modeling. Although there is no obviously different from Model 1, overall, Model 2 performs worse than Model 1. Furthermore, Model 3 based on preprocessing dataset with normalisation only. Compared with Model 1, it rescales all predictors (mean=0, deviation=1). Although there is no obviously different with Model 1 on RMSE. But Model 3 makes MAD and NLL even worst. Lastly, things went even worse on Model 4, which based on preprocessing dataset combined with feature selection and normalisation which are mentioned in Model 2 and Model 3 above.

So far, we could find both normalisation and feature selection could not improve the linear regression model performance and they all shrink MAD which means the model would cover less lager fire area. To fix the unbalanced problem on the target field 'area', SOMTE[14] is implemented by Model 5. As shown below, by SOMTE, the number of examples in each area group is the same as each other.

[14] ZHONG LS, GAO XJ, WANG ZY. A New Kind of Improving SOMTE Algorithm Based on K-means in Imbalanced Datasets. Mathematics in Practice and Theory. 2015;2015(19):26.

```
[37] #  target value distribution in original data
     train.area_bins.value_counts()
```

```
0    253
2     49
1     39
3      9
5      6
4      2
Name: area_bins, dtype: int64
```

```
# by up-sampling
train_up.area_bins.value_counts()
```

```
5    100
4    100
3    100
2    100
1    100
0    100
Name: area_bins, dtype: int64
```

However, the model performance did not improve by the balanced distribution training data. This strategy improves MAD and reduces the value of NLL, it achieved the worst prediction with quite highest in RMSE and lowest on AUC as shown below:

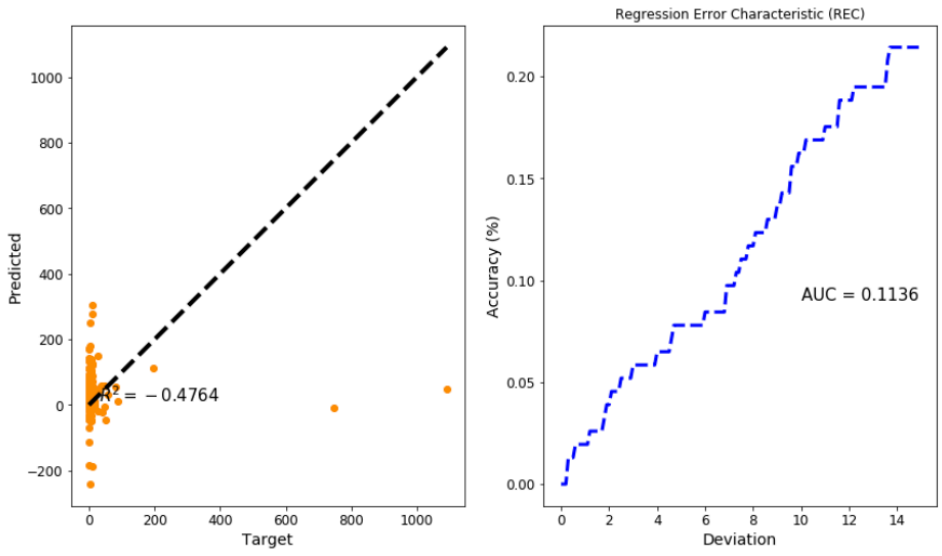


Figure 19, 20. Model Evaluation - Lasso Regression-2

The main issue with 'Random Up-Sampling'[15] is that we run the risk that our classification models will not perform as accurate as expected since there is a great deal of information loss (bringing only 100 non-fire area from 253 non-fire area samples randomly) and there is little information for up-sampling in group3, group5, and group4(They have only 9 instances, 6 instances, and 2 instances respectively in training set).

The experiments were run on :
Hardware - Processor : 2.6GHz Intel Core i7
Memory : 16GB 2400MHz DDR4,
Graphics : Radeon Pro 560X 4GB, Intel UHD Graphics 630 1536MB Software - OS : MacOS Mojave 10.14.6
Python 3.7.3 (default, Mar 27 2019, 16:54:48), IPython 7.6.1, Jupyter Notebook 6.0.0, Sklearn version 0.21.3

[15] Visa S, Ralescu A. Issues in mining imbalanced data sets-a review paper. InProceedings of the sixteen midwest artificial intelligence and cognitive science conference 2005 Apr 16 (Vol. 2005, pp. 67-73). sn.

5. Conclusion

- Overall, among Random Forest regression, Decision Tree regression and Lasso regression, Random Forest regression achieved the best performance by the logarithm without upsampling. Whereas, Lasso regression worked better to predict small fire areas.

- Since we faced the unbalanced data, we computed a number of pre-processing steps such as encoding, scaling, logarithm, and feature selection. However, the result of each of them depends on what model you choose. There is no guarantee to make the model performance improve when you apply one of these skills or a combination of some of them.

- In this project, the performance of scaling and feature selection did not appear obvious. On the other hand, those methods might be more generally used for the majority of the general modeling processes.

- The evaluation indicators of regression are not limited just for MSE, RMSE or MAE, but we could widen the scope of methods, MAD and AUC, for example. We expect that finding out the appropriate evaluation method is another important step for the prediction project.

- 'X' and 'Y' features in our data set represent the x-axis and y-axis spatial coordinate within the Montesinho park map, respectively. We assume that not only meteorological data would affect the fire, but also the geometric data may affect as well. Specifically, it is quite obvious that the river or the lake could prevent the big fire but the dry area definitely makes the burned area bigger, on the other hand. Hence, we would suggest using geometric data such as 'X' and 'Y' for finding out another remarkable correlation further.

- As you can see the boxplot below, we could say that only a few incidents happened in December but all the burned area in that period seems relatively wider than others. From this data investigation, we would suggest to find out reasonable opinions could support which attributes possibly impact on this situation.

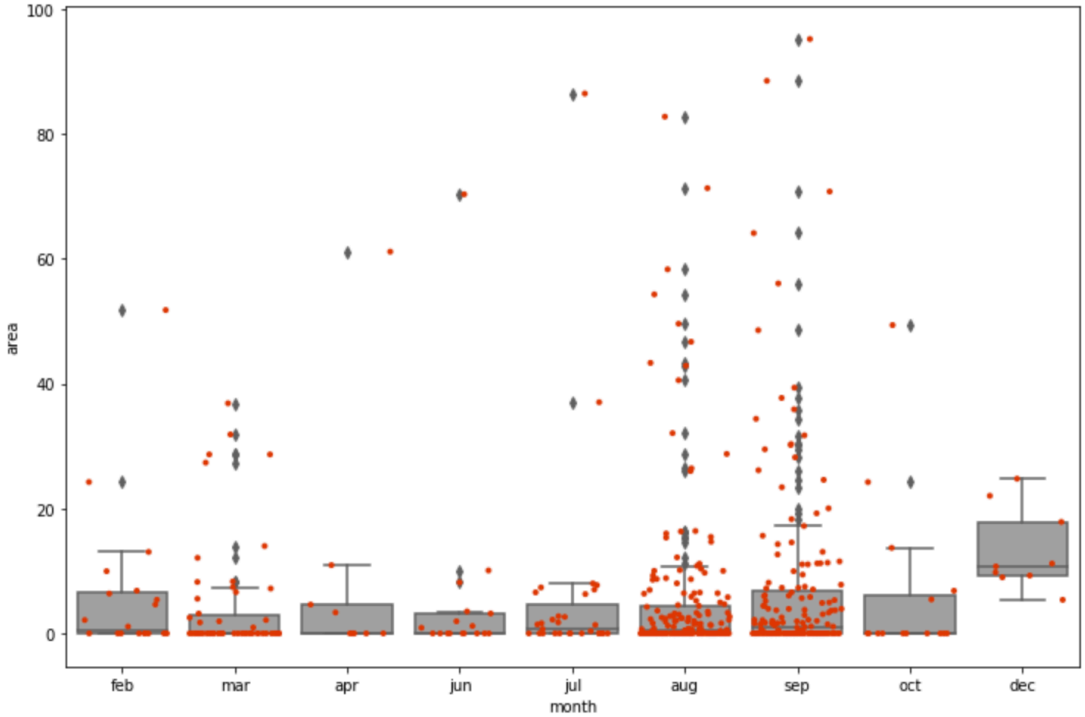


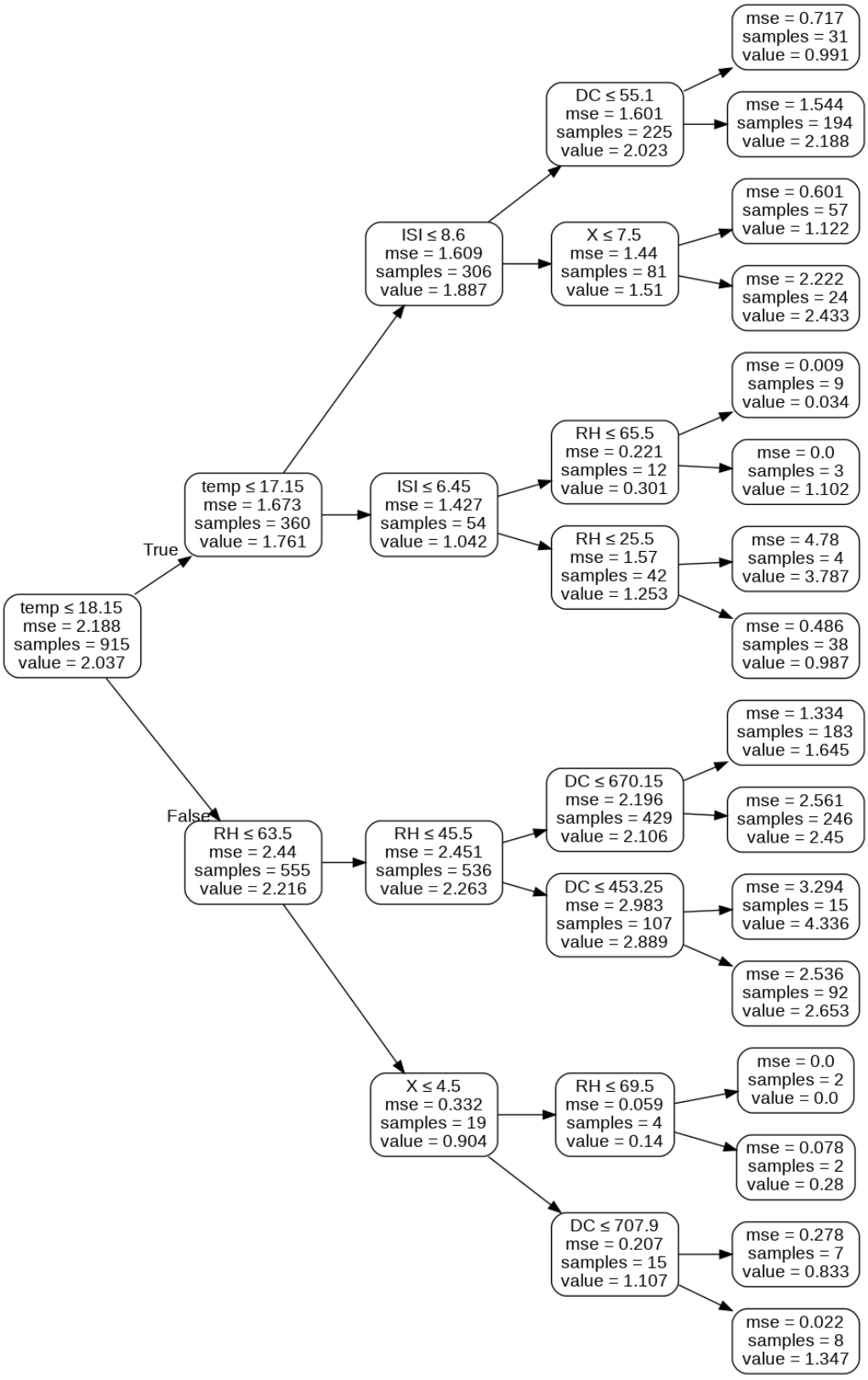
Figure 21. The relationship between months and the size of burned area

6. References

1. Cortez P, Morais Guimar˜aes, Portugal A. A Data Mining Approach to Predict Forest Fires Paulo Cortezusing Meteorological Data. Guimaraes, Portugal;
2. Nguyen HT. [Unpublished lecture notes on Machine Learning and Data Mining]. University of Sydney; notes provided at lecture 2019 Oct 2.
3. Ahiale D. Predict The Burned Area Of Forest Fires. [Kaggle online notes] 2017 Dec 8
4. Abeel T, Helleputte T, Van de Peer Y, Dupont P, Saeys Y. Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. Bioinformatics. 2009 Nov 25;26(3):392-8.
5. Kotsiantis SB. Decision trees: a recent overview. Artificial Intelligence Review. 2013 Apr 1;39(4):261-83.
6. Bi J, Bennett KP. Regression error characteristic curves. InProceedings of the 20th international conference on machine learning (ICML-03) 2003 (pp. 43-50).
7. Mohtadi BF. In depth parameter tuning for random forest. Published on All thing in AI. 2017 Dec 22.
8. Ahiale D. Predict The Burned Area Of Forest Fires. [Kaggle online notes] 2017 Dec 8
9. 3. 3.1. Cross-validation: evaluating estimator performance — scikit-learn 0.21.3 documentation [Internet]. Scikit-learn.org. [cited 1 November 2019]. Available from: https://scikit-learn.org/stable/modules/cross_validation.html
10. [Internet]. [cited 2 November 2019]. Available from: https://scikit-learn.org/stable/modules/cross_validation.html
11. Pokrass J, Bronstein AM, Bronstein MM, Sprechmann P, Sapiro G. Sparse modeling of intrinsic correspondences. InComputer Graphics Forum 2013 May (Vol. 32, No. 2pt4, pp. 459-468). Oxford, UK: Blackwell Publishing Ltd.
12. Ahiale D. Predict The Burned Area Of Forest Fires. [Kaggle online notes] 2017 Dec 8
13. TravelCodeSleep. End-to-End Regression Pipeline Using ScikitLearn. [Kaggle online notes] 2019 Jan
14. ZHONG LS, GAO XJ, WANG ZY. A New Kind of Improving SOMTE Algorithm Based on K-means in Imbalanced Datasets. Mathematics in Practice and Theory. 2015;2015(19):26.
15. Visa S, Ralescu A. Issues in mining imbalanced data sets-a review paper. InProceedings of the sixteen midwest artificial intelligence and cognitive science conference 2005 Apr 16 (Vol. 2005, pp. 67-73). sn.

7. Appendix

7.1 Decison Tree for the regression structure



7.2 How to run the code

We separated the code into three different files containing one algorithm each. Our three regressors were maintained in each own ipynb file under the required naming convention, and each regressor stays in one code cell. The best model is RandomForestRegressor and the other two models are DecisionTreeRegressor and Lasso Regressor.
By default, these codes read the data file named “forestfires.csv” under the current directory or under “./data” directory. If the code can not found in this file in both directories, please download the data set from Haichen’s Github repository, and corresponding information will be printed out. To run the code, each regressor will be automatically trained and tested on a test data set, and the performance will be print out under the code with just one click.
The packages we used in our code are mainly common packages of Python, including math, numpy, scipy, pandas, matplotlib, time seaborn and sklearn.

7.3 Contribution

Member	Contribution
Haichen Zhu	Pre-Processing + Pre-Processing + Decision Tree + Reference + Code Gathering
Yeseul Yoon	Introduction + Pre-Processing + Random Forest + Conclusion + Report Gathering
Kuo Yuan	Abstract + Pre-Processing + Lasso Regression + Conclusion