Data analysis and prediction about Beijing pm2.5

Niu Shengjie, Wu Yifan

May 30, 2022

Abstract

Beijing, the capital of China, where people are suffer from severe pollution caused by pm2.5, therefore, we urgently need to analyze and predict pm2.5 content in Beijing, which can not only provide help for pm2.5 prevention and control, but also give people an intuitive reference of pm2.5 level every day. In this paper, we analyze pm2.5 data based on the data of pm2.5 from 2010 to 2014 in three major aspects: regression, classification and time series analysis. We will show that in the aspects of the goodness of fit of the regression model, the accuracy of the classification model and the prediction results of the time series analysis, there will be satisfactory results.

Keywords: PM2.5; Regression; Classifcation; Time Series Analysis; Feature importance; Missing data imputation

Contents

1 Introduction

Now adays, Most of China are experiencing chronic air pollution. The main pollutants are fine particulate matter, and pm2.5 in particular. pm2.5 refers to particles with aerodynamic equivalent diameter less than or equal to 2.5 microns in the ambient air. It can be suspended in the air for a long time. The higher its concentration in the air, the more serious the air pollution is. Although pm2.5 is only a small component in the earth's atmosphere, but it has an important impact on air quality and visibility. Compared with coarse atmospheric particles, pm2.5 small particle size, large area, strong activity, easy to carry toxic and harmful substances (such as heavy metals, microorganisms, etc.), long residence time in the atmosphere and long transportation distance.

Beijing, the capital of China, has a high pm2.5 index. As the same time, pm2.5 also causes inevitable damage to Beijing and Beijing people. When inhaled, pm2.5 gets into the bronchi, where it interferes with gas exchange in the lungs and causes diseases including asthma, bronchitis and cardiovascular disease. In addition, these particles can also enter the blood through the bronchi and alveoli, where harmful gases, heavy metals and other dissolved in the blood, the harm to human health is greater. Therefore, the assessment and monitoring of pm2.5 in Beijing is of great importance and of particular interests.

In this case, we not only need to find appropriate and effective ways to reduce the content of pm2.5 in the air, so as to reduce its harm to people; We should also analyze the distribution and law of pm2.5 and find out all the factors that affect pm2.5 content, such as time, temperature, humidity, wind direction, atmospheric pressure and so on, so that we can better understand the law of pm2.5 generation and predict the content of pm2.5 in the future. At the same time, pm2.5 can be classified according to the degree of harm to people, so that people can choose the right time to go out. Therefore, it is necessary to analyze the historical data of pm2.5.

2 Exploratory Data Analysis

2.1 Data Description

First let us take a general understanding of the overall data set, the dataset gives us hourly pm2.5 levels in Beijing from 2010 to 2014, which is the dependent variable. As well as hourly data on other influencing factors, namely, independent variable:

•**No** : Row number

•**year** : Year of data in this row

•**month** : Month of data in this row

•**day** : Day of data in this row

•**hour** : Hour of data in this row

•**DEWP** : Dew Point

•**TEMP** : Temperature

•**PRES** : Pressure (hPa)

•**cbwd** : Combined wind direction

•**lws** : Cumulated wind speed

•**ls** : Cumulated hours of snow

•**lr** : Cumulated hours of rain

Therefore, we use these known information to analyze and excavate the regularity and intrinsic information of pm2.5 content in Beijing.

2.2 Statistical data analysis

After description, we need to analyze the data from a statistical perspective. First, we drew a continuous bar chart to show the proportion of pm2.5 in each grade from 2010 to 2014, the graph is as following :

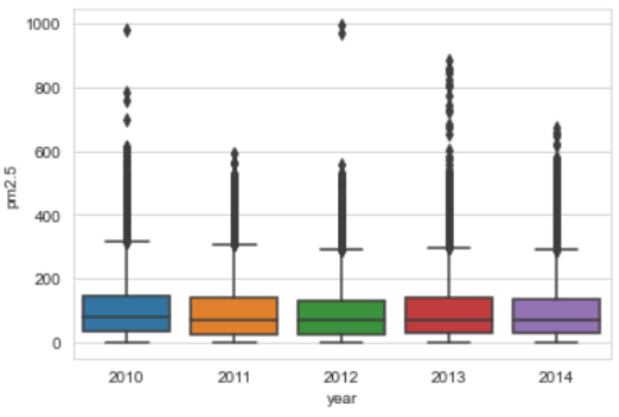


Figure 1: Boxplots of pm2.5 in five years

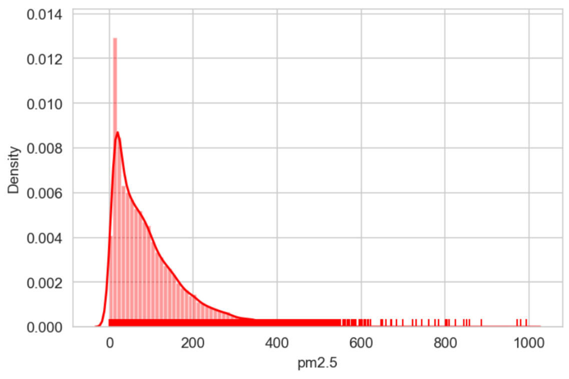


Figure 2: Distribution plot of pm2.5

As we can see, most of the data ranges from 10 to 200 and when pm2.5 values rose, so did the numberof outliers. I think this fits our intuition well, because pm2.5 doesn't go below zero. At the same time, we found that the data of pm2.5 is obviously skewed, which reminds us that we may need to make some transformations on the response variable pm2.5.

And we tried to find the relationship between other types of variables and pm2.5 such as month and hours, the boxplots of pm2.5 based on other variables are given below:

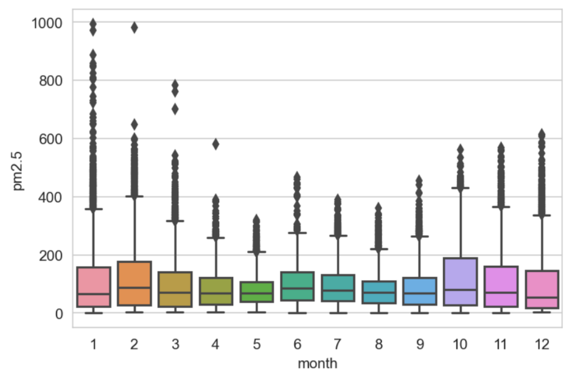


Figure 3: Boxplots of pm2.5 per month

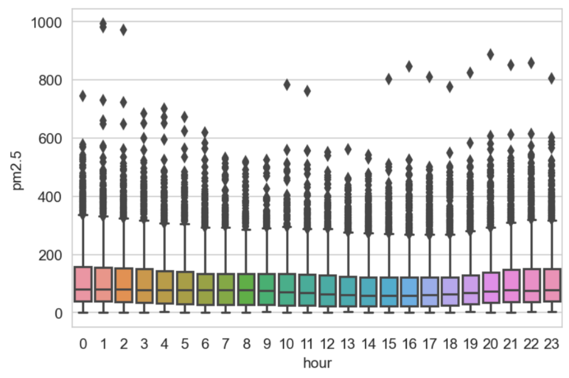
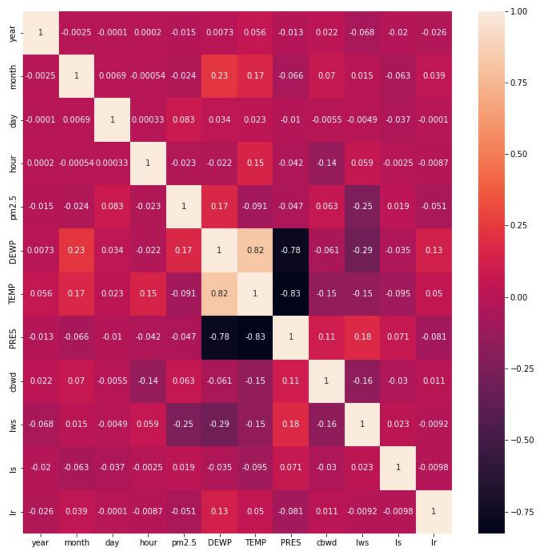


Figure 4: Boxplots of pm2.5 per hour

From above figures, we found that pm2.5 levels were generally higher in winter than in summer, this discovery led us to plan to mine more information about the event to analyze our data. However, the distribution of pm2.5 in each hour is very similar so we will discuss the hours further in the later section.

At last, we want to know that if there was a correlation between each of the variables, so we calculated the correlation coefficient between each of the variables, And then the heatmap plot is given below:



	year	month	day	hour	pm2.5	DEWP	TEMP	PRES	cbwd	lws	ls	lr
year	1	-0.0025	-0.0001	0.0002	-0.015	0.0073	0.056	-0.013	0.022	-0.068	-0.02	-0.026
month	-0.0025	1	0.0069	-0.00054	-0.024	0.23	0.17	-0.066	0.07	0.015	-0.063	0.039
day	-0.0001	0.0069	1	0.00033	0.083	0.034	0.023	-0.01	-0.0055	-0.0049	-0.037	-0.0001
hour	0.0002	-0.00054	0.00033	1	-0.023	-0.022	0.15	-0.042	-0.14	0.059	-0.0025	-0.0087
pm2.5	-0.015	-0.024	0.083	-0.023	1	0.17	-0.091	-0.047	0.063	-0.25	0.019	-0.051
DEWP	0.0073	0.23	0.034	-0.022	0.17	1	0.82	-0.78	-0.061	-0.29	-0.035	0.13
TEMP	0.056	0.17	0.023	0.15	-0.091	0.82	1	-0.83	-0.15	-0.15	-0.095	0.05
PRES	-0.013	-0.066	-0.01	-0.042	-0.047	-0.78	-0.83	1	0.11	0.18	0.071	-0.081
cbwd	0.022	0.07	-0.0055	-0.14	0.063	-0.061	-0.15	0.11	1	-0.16	-0.03	0.011
lws	-0.068	0.015	-0.0049	0.059	-0.25	-0.29	-0.15	0.18	-0.16	1	0.023	-0.0092
ls	-0.02	-0.063	-0.037	-0.0025	0.019	-0.035	-0.095	0.071	-0.03	0.023	1	-0.0098
lr	-0.026	0.039	-0.0001	-0.0087	-0.051	0.13	0.05	-0.081	0.011	-0.0092	-0.0098	1

Figure 5: Heatmap plot

We find that only the correlation coefficient between TEMP(temperature) and DEWP(dew point) is larger, which is 0.82. And the response variable pm2.5 is only partially correlated with DEWP, with a correlation coefficient of 0.17. In addition, some pairs of variables have very small correlations. The correlations between DEWP and month are 0.23, lws and PRES are 0.18, TEMP and month are 0.17, TEMP and hour are 0.15, lr and DEWP is 0.13, cbwd and PRES is 0.11, while the other pairs of variables have little relationship to each other, which can be ignored.

According to common sense, there is a certain correlation between wind direction and PM2.5, so we only calculate the correlation coefficient between wind direction and PM2.5, and get the following results

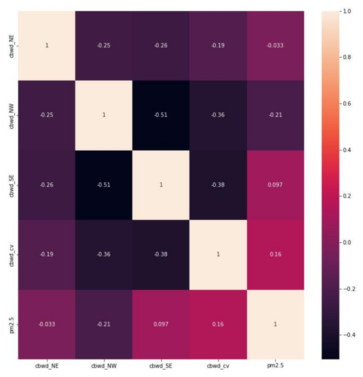


Figure 6: Heatmap only wind

From the above figure, we can conclude that there was a weak correlation between CV and PM2.5, with a correlation coefficient of 0.16. The correlation between SE and PM2.5 was even weaker, with just a correlation coefficient of 0.097. While the other two wind directions has no correlation with PM2.5.

2.3 Classification of pollution levels

We hope to find an indicator to evaluate the pollution level of PM2.5. At the same time, for the convenience of subsequent research, we hope to classify the content of pm2.5. According to the US (EPA) standards, 35 $ug\ m^{-3}$ is the maximum acceptable level of pm2.5, while it is harmful to humans at levels above 150 $ug\ m^{-3}$. Thus, we have divided the content of pm2.5 into three levels : low state when pm2.5 smaller than 35 $mg\ m^{-3}$, polluting episode when pm2.5 between 35 $ug\ m^{-3}$ and 150 $ug\ m^{-3}$, and even higher when pm2.5 bigger than 150 $ug\ m^{-3}$. To reduce the noise in the PM readings, we smooth the time series over 3 hours moving windows. After classifying pm2.5, we drew a bar chart of pm2.5 about total years:

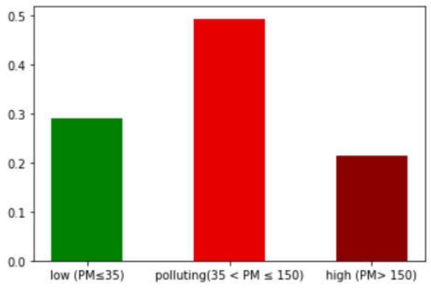


Figure 7: Distribution of different levels pm2.5

By analysing the result, we found that, polluting episode occupied nearly half of the total time, only less than 30 percent of the time is in the acceptable level. More seriously, people are harmed by PM2.5 nearly 20 percent of the year. Then, we draw the distribution graph separately for every year :

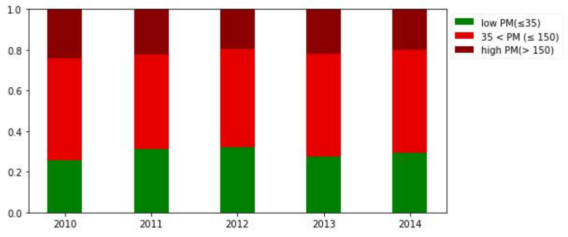


Figure 8: Partition of different levels pm2.5 per year

What we find is that the distribution in each of these five years is very consistent with the overall distribution and these percentages and statistics did not vary significantly over the 5 years. All these statistics portray a very severe situation for Beijing's air pollution. Therefore, we hope to build a classification model so that people can decide whether to go out or not based on the weather conditions on that day.(section 8: **Classification**)

3 Data Preprocessing

3.1 Missing Data

A very important part of data preprocessing is checking for missing values and dealing with missing. We need to look at the case of missing values first, there we use method: *isna().sum()* to check the number of missing values in each column. The corresponding output are given below:

```
year                   0
month                  0
day                    0
hour                   0
pm2.5               2067
DEWP                   0
TEMP                   0
...
cbwd_NE                0
cbwd_NW                0
cbwd_SE                0
cbwd_cv                0
```

We find that only the "pm25" column of data has null values, a total of 2067 rows. The total data set has a total of 43,824 rows, and the data containing null values accounts for a small proportion of the total data set. In this report, we take different approaches for different tasks, which contains direct deletion and several filling methods.

3.1.1 Delete the Missing Data

From the above, it can be known that the data containing null values accounts for a small proportion of the total data set and can be deleted directly.

Here we use the method *dropna()* to directly remove observations with missing values and get a dataset that can be used for analysis. The analysis process for the data after deleting missing values will be shown in detail in the section 4 (**Regression**) and section 5 (**Data Addition and Regression**).

3.1.2 Filling the Missing Data

For observations with missing data, in addition to directly deleting the observations, we can also fill in the missing values. The filling method used and the analysis of the filled data will be explained in detail in the section 6 (**Data Filling and Regression**).

3.2 Data Conversion

3.2.1 One-hot Coding

In the dataset, we encountered categorical feature: "cbwd", which has four types: "NE", "NW", "SE" and "cv", these feature values are not continuous, but discrete and disordered.

To use it as an input for machine learning, we need to characterize it numerically, here we use one-hot coding by using an N-bit state register to encode N states, each state has its Independent register bits, and only one of them is valid at any one time.

We use the method *get-dummies* in the *pandas* to realize this conversion.

3.2.2 Data Scaling

In the process of data analysis and prediction, we want to compare the importance of each feature in predicting the dependent variable, but different features have the different units, so we need to standardize these features, which is important for our subsequent analysis of Importance of each feature.

Here we use Z-score scaling, i.e., $x_i^* = \frac{x_i - \hat{\mu}}{\hat{\sigma}}$ where $\hat{\mu}$ is the sample mean and the $\hat{\sigma}$ is the sample variance. And we use method *preprocessing.scale()* in *sklearn* to realize it.

3.2.3 Transformation of Response Variable

In the section 2 (**Exploratory Data Analysis**), by observing the distribution image of the dependent variable "pm2.5", we will find that there are too many outliers and most of which are points with large values. Therefore, we consider reducing the scale of the dependent variables by making some changes to them. One of the most common scaling transformations is log transformation. The following is the distribution image of the dependent variable "pm2.5" after log transformation.

We can find that the number of outliers experiences a substantial reduction after log transformation.

In addition, we also found that the model obtained by linear regression directly with the original data does not satisfy the normality assumption, and the goodness of fit (R^2 =) is not high. Using the log-transformed response variable for fitting can significantly improve the goodness of fit (R^2 =), and the newly obtained model can satisfy the normality assumption (Figure). Therefore, in most of this report, we use the log transformed "pm2.5" as the new response variable.

3.3 Outliers Detection

We believe that it is unreasonable to judge and detect outliers only by statistics based methods like box plot.

Here we use local outlier factor method to detect the outliers, which is a density based method. We could compute the density at each position $\boldsymbol{x}$, and compare the density of each point $\boldsymbol{x}$ with the density of its neighbors. The core idea is to judge the degree of abnormality by the local environment of given point.

Here we implement this method by using the model function *LocalOutlierFactor* from the *sklearn* package.

We set the value of the abnormal degree in the top 5% as outliers, and delete these observations that we think are unreasonable.

3.4 Split Training and Test Sets

In machine learning, we need to split the data set into a test set and a training set, and we do not use random divisions here. We need to split the remaining data to training and test sets by following this rule: extract the items (rows) for every 7-day from 2010-01-07 to 2014-12-25.

After our observations, we found that the criteria for selecting the test set was to select the data of each Thursday. So we first combine the features: "year", "month" and "day" of each observation into the form of *datetime*. Then we can use the method *weekday* in *datetime* to judge whether the observation is Thursday's data. After that we pick out all observations in Thursday and use it as the test set and the rest of the data as the training set.

4 Regression

4.1 Ordinary Least Squares

We first consider the most common method, simple multiple linear regression. Here we first use meteorological indices DEWP, TEMP, PRES, cbwd (have been processed by one-hot coding), lws, ls, and lr, as covariate vector X, and treat pm2.5 value as the response Y.

Then we use the covariate vector to fit the response at first. we can get the linear regression model with $R^2 = 0.258$. Also, each predict variable(t-value) and F-statistic are significant, so we can think this regression equation is meanfuling. However, there are many problems with this model. Without considering the collinearity verified in the previous section, we will first consider verifying several assumptions of linear regression.

Normality

Q-Q plot is also known as Quantile-Quantile Plots, is a probability plot method for comparing two probability distributions by comparing their quantiles. And if the error obey the normal distribution, the Q-Q plot should appear as a straight line.

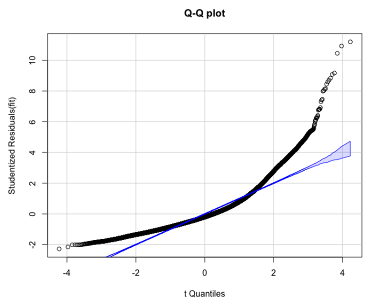


Figure 9: Q-Q plot of OLS without variable transformation.

From the Q-Q plot, we can get that the regression model does not satisfy the normality. In this case we always prioritize some transformation of the response variable, here we use the boxcox normal transformation. We can normalize the variable Y^λ by the maximum likelihood estimation of λ, and the figure is given below:

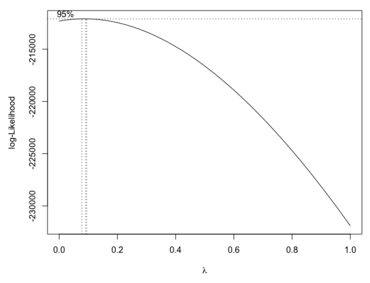


Figure 10: log-likelihood curve without variable transformation.

From this figure, we have that the log-likehood reach the maximum when the λ is nearly 0.1, which is far away from 1, so we consider take the *box-cox* transformation of the response varibale. Then we can get the new log-likehood figure and Q-Q plot.

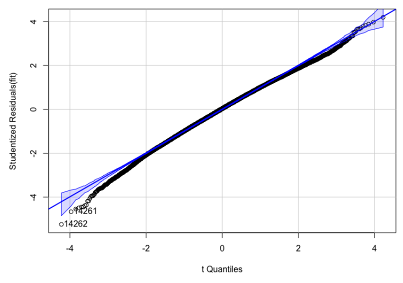


Figure 11: Q-Q plot of OLS with Box-cox transformation of response variable.

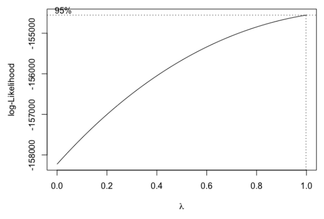


Figure 12: log-likelihood curve with variable transformation.

According to the figure 4 and figure 5, we have that the new regression model satisfy the normality assumption. Then we consider the new linear regression model by using the new response variable after Box-cox transformation. Using the same covariate X to generate model2. We have the generated model 2 has the $R^2 = 0.428791$ in test set and this model satisfy the normality assumption. Then we should consider the other assumptions.

Linearity

We can look for evidence of nonlinearity in the relationship between the dependent variable and the independent variables by using component plus residual plots (also known as partial residual plots). We're looking for any systematic departure from the linear model that we've specified.

To create a component plus residual plot for variable X, we need to plot the points: $\epsilon_i + (\hat{\beta}_0 + \hat{\beta}_1 \cdot X_{1i} + \cdots + \hat{\beta}_k \cdot X_{ki})$ vs. X_i. where the residuals are based on the full model, and $i = 1, 2, \cdots, n$. The straight line in each graph is given by $(\hat{\beta}_0 + \hat{\beta}_1 \cdot X_{1i} + \cdots + \hat{\beta}_k \cdot X_{ki})$ vs. X_i. Nonlinearity in any of these plots suggests that you may not have adequately modeled the functional form of that predictor in the regression. The resulting plots are provided in figure 6.

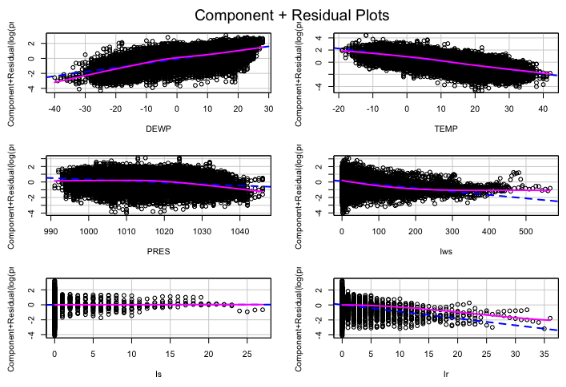


Figure 13: Component plus residual plots for the regression of pm2.5.

The component plus residual plots confirm that you've met the linearity assumption. The form of the linear model seems to be appropriate for this dataset.

Homoscendity

The *ncvTest()* function produces a score test of the hypothesis of constant error variance against the alternative that the error variance changes with the level of the fitted values. A significant result suggests heteroscedasticity (nonconstant error variance). The result of *ncvTest()* are given below:

```
> ncvTest(fit)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 144.9775, Df = 1, p = < 2.22e-16
```

The non-constant variance score test is nonsignificant ($p \leq 2.22e^{-16}$), suggesting that this regression can not met the constant variance assumption. We can also see that in the spread-level plot (figure 7), we see a non-horizontal curve. If this regression meet the assumption, we'd expect to see that the points form a random horizontal band around a horizontal line of best fit.

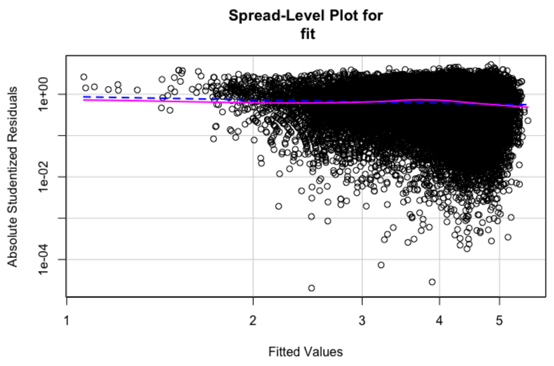


Figure 14: Spread-level plot for assessing constant error variance.

Independence

The best way to assess whether the dependent variable values (and thus the residuals) are independent is from your knowledge of how the data were collected. For example, time series data often display autocorrelation—observations collected closer in time are more correlated with each other than with observations distant in time. In this dataset, the response variable (pm2.5) is time series concerning the year, month, and day.

We can use Durbin–Watson test to detect such serially correlated errors. Here we can apply the Durbin–Watson test to the multiple-regression problem and the result are given below:

```
> durbinWatsonTest(fit)
> lag Autocorrelation D-W Statistic p-value
   1    0.8861811      0.2276367      0
   Alternative hypothesis: rho != 0
```

The significant p-value (p=0) suggests the existence of autocorrelation and, conversely, an dependence of errors. However, other three assumptions are satisfied, we can also use these linear regression model to predict the future pm2.5.

4.2 Multicolinearity and Ridge Regression

4.2.1 Detection of Multicolinearity

Multicollinearity in regression refers to the case when one predictor variable in a multiple regression model can be linearly predicted from the others with a substantial degree of accuracy.

In the section 2 (**Exploratory data analysis**), we can find that there exist the significant correlations between pairs of independent variables in the dataset, So we have good reason to check for multicollinearity.

We can check the tolerance or the variance inflation factor (tolerance < 0.1 or VIF > 10) to detect the multicollinearity.

Table 1: VIF and tolerance for each features

Features	**DEWP**	**TEMP**	**PRES**	**Iws**
VIF	4.047	4.706	3.663	1.263
Tolerance	0.247	0.213	0.273	0.792
Features	**ls**	**lr**	**cbwd_NE**	**cbwd_NW**
VIF	1.014	1.039	4377.095	11926.480
Tolerance	0.986	0.962	0.0002	8.385e-05
Features	**cbwd_SE**	**cbwd_cv**		
VIF	13226.487	8273.583		
Tolerance	7.56e-05	0.00012		

And we can see that multicollinearity exists for 'cbwd-NE', 'cbwd-NW', 'cbwd-SE' and 'cbwd-cv'. Existence of multicollinearity will lead some terrible consequences: (1). Difficult to test individual regression coefficients due to inflated standard errors. (2). Unstable coefficient estimates, sensitive to small change in the model.

Fortunately we have some remedial measures to alleviate the influence of consequences: (1). Shrinkage methods such as ridge regression (2). Drop one or several highly correlated independent variables (or by stepwise regression to select appropriate variables) (3). Combine variables (dimension reduction by the use of methods such as principle component procedures).

In this part, we first try to solve this problem with ridge regression, other approaches will be discussed later.

4.2.2 Ridge Regression

Ridge regression is a biased estimation regression method dedicated to the analysis of collinear data. It is essentially an improved least squares estimation method. By giving up the unbiasedness of the least squares method, the regression is obtained at the expense of losing some information and reducing accuracy. The coefficients are more realistic and reliable regression methods, and the fit to ill-conditioned data is stronger than the least squares method.

In practice, it is necessary to constantly adjust the regular parameter Alpha to seek a balance in the above process. RidgeCV implements cross-validation of ridge regression. The following is an efficient cross-validation method: leave-one-out and find that the best choice of parameter α is 14.22.

Next we can evaluate the accuracy of this model, we find that the R square for train set and test set are 0.408527 and 0.428777 respectively, what is more, the MSE of test set are 1.864496.

Now that we have finished learning the model for ridge regression, let's do some comparisons with the linear fit model. The conseuences is given below:

Table 2: Comparison between linear and ridge model

Model	R^2 **(train)**	R^2 **(test)**	**MSE (test)**
Linear Model	0.408527	0.428791	1.864451
Ridge Model	0.408527	0.428777	1.864496

It can be seen from the table 2 that linear regression and ridge regression work almost the same on the training set, but the effect on the test set is slightly effective than that of ridge regression, indicating that the generalization ability of linear regression is stronger. But in general, the difference between them is minimal, this is because Our dataset is large enough that the information they get is not very different.

4.3 Ensemble Method of Linear Regression

4.3.1 Boosting

The main idea of boosting is to assemble weak classifiers into a strong classifier. Boosting is a combination of weak learners based on weights. It adopts an iterative algorithm. Each iteration weights the samples according to the prediction results of the previous iteration. Therefore, as the iteration continues, the error will become smaller and smaller, and it will become closer to the truth value.

Now we take *Adaboost* method, short for Adaptive Boosting, which is a statistical classification meta-algorithm formulated by Yoav Freund and Robert Schapire, who won the 2003 Gödel Prize for their work. *Adaboost* is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms. *Adaboost* is a very popular boosting technique that aims at combining multiple weak classifiers to build one strong classifier.We use *Adaboost* for linear regression and ridge regression separately. Here we adjust n-estimator = 400 and learning-rate = 0.001. The conseuences is given in the following table:

Table 3: Comparison between linear, ridge and the version of adaboost

Model	R^2 **(train)**	R^2 **(test)**	**MSE (test)**
Linear Model	0.408527	0.428791	1.864451
Ridge Model	0.408527	0.428777	1.864496
Linear(Adaboost)	0.408439	0.428655	1.864892
Ridge(Adaboost)	0.408448	0.428658	1.864883

4.3.2 Bagging

For a certain data set, through random sampling T times, we can get T sampling sets. For these T sampling sets, we can train T weak learners independently, and then we can train the T weak learners independently. Weak learners are combined into strong learners.

It should be noted that each time the data is randomly collected, there are T times of replacement and collection, and finally T different sample sets are obtained.

Now we take *Bagging* method for linear regression and ridge regression separately. Here we adjust n-estimators=1000, max-samples=0.7 and max-features=1.0. The conseuences is given in the following table:

Table 4: Comparison between linear, ridge and the version of bagging

Model	R^2 **(train)**	R^2 **(test)**	**MSE (test)**
Linear Model	0.408527	0.428791	1.864451
Ridge Model	0.408527	0.428777	1.864496
Linear(bagging)	0.408527	0.428772	1.864512
Ridge(bagging)	0.408527	0.428770	1.864517

We found that neither bagging nor adaboost improved our model, and even the goodness of fit would drop slightly.

4.4 Other boosting algorithms

Since the above model does not fit the data very well, so we try to use other ensemble learning model, called *Decision Tree*, to analyze the data. Decision tree is a type of supervised machine learning used to categorize or make prediction based on how the previous set of questions were answered.

The decision tree predicts the result based on one or more input variables and combined with the division conditions. The splitting process starts from the root node of the decision tree: at each node, the algorithm will check whether the input variable needs to be divided intermittently to the left cotyledon and the right cotyledon according to the division conditions. Stop splitting when a child node (end point) of any decision tree is reached.

In this part, we use *GBDT*, *XGBoost* and *LightGBM* to analyze the data. It is found that the prediction effect of these ensemble learning models are significantly higher than that of the above model. And at the same time, we still have opportunities to further improve the fitting effect of the model

4.4.1 GBDT

GBDT (Gradient Boosting Decision Tree) works well in data analysis and prediction. It is an ensemble algorithm based on decision tree. Among them, Gradient Boosting is an algorithm in the integrated method boosting, which iterates the new learner through gradient descent. We use CART (regression tree with gain with gini index) as base classifier, It can handle various types of data flexibly, and can use some robust loss functions, which is more robust to outliers.

Here, we adjust n-estimator = 400 and learning-rate = 0.1, we will have the following consequence and prediction image in test set.

```
R^2 in train set:0.553716
R^2 in test set:0.554174
MSE in test set:1.335932
```

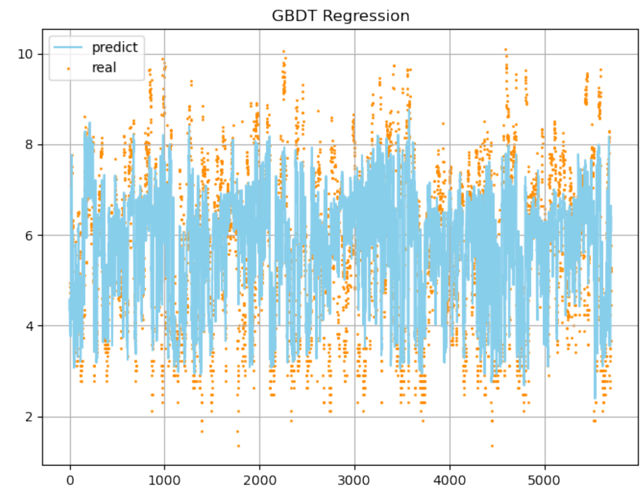


Figure 15: Predicted figures of GBDT

4.4.2 XGBoost

XGBoost can be regarded as an implementation of an upgraded version of the GBDT algorithm (second-order Taylor expansion is performed, and second-order gradient information is used).

XGBoost adds a regular term to the cost function to control the complexity of the model. The regular term includes the number of leaf nodes of the tree, etc. The regular term reduces the variance of the model, making the learned model simpler and preventing overfitting, which is also a feature of XGBoost superior to traditional GBDT.In addition, XGBoost can also carry out parallel learning.

Here, we adjust max-depth = 5, n-estimator = 400 and learning-rate = 0.1, we will have the following consequence and prediction image in test set.

```
R^2 in train set:0.610723
R^2 in test set:0.563186
MSE in test set:1.308926
```

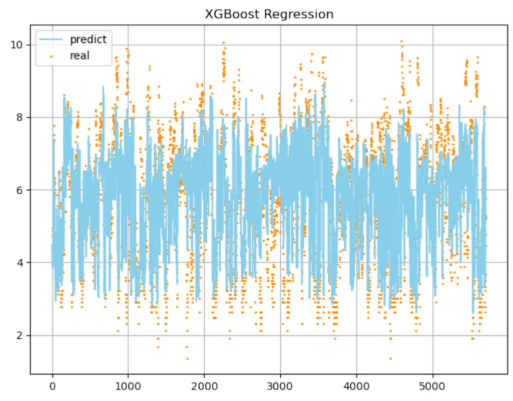


Figure 16: Predicted figures of XGBoost

4.4.3 LightGBM

LghtGBM is actually an implementation of the GBDT method, which use some optimizations for the same goal.

Different from the XGboost algorithm that adopts the level-wise strategy, it adopts the optimal leaf-wise strategy to split leaf nodes. Therefore, in the LightGBM algorithm, when growing to the same leaf node, the leaf-wise algorithm reduces more loss than the level-wise algorithm. This results in higher accuracy than any other existing boosting algorithm can achieve.

Here, we adjust n-levels = 31, n-estimator = 1000 and learning-rate = 0.01, we will have the following consequence and prediction image in test set.

```
R^2 in train set:0.577344
R^2 in test set:0.559329
MSE in test set:1.320484
```

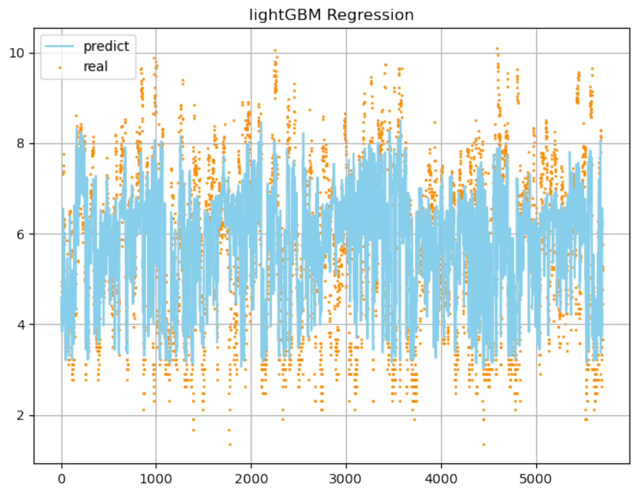


Figure 17: Predicted figures of LightGBM

4.5 Summary

In this section, we first used simple linear regression, and then in order to solve the problem of collinearity, we used ridge and lasso to shrink the model, and after that, we applied bagging and boosting methods to these Linear models, but such results are still not good enough. Therefore, we used the GBDT algorithm and the two optimization algorithms based on GDBT (XGBoost and LightGBM) to perform model fitting again, and achieved a relatively good fitting effect.

The fitting effects obtained in this section are summarized in the table below:

Table 5: Summary of different regression methods

Model	**R^2 (train)**	**R^2 (test)**	**MSE (test)**
Linear Model	0.408527	0.428791	1.864451
Ridge Model	0.408527	0.428777	1.864496
Linear(bagging)	0.408527	0.428772	1.864512
Ridge(bagging)	0.408527	0.428770	1.864517
Linear(bagging)	0.408527	0.428772	1.864512
Ridge(bagging)	0.408527	0.428770	1.864517
GBDT	0.553716	0.554174	1.335932
XGBoost	0.610723	0.563186	1.308926
LightGBM	0.577344	0.559329	1.320484

Compared with linear regression, ridge regression and their ensemble versions, these algorithms based on decision trees significantly improve the model's goodness of fit.

At this point we still have lots of exploratory data not added, and outliers not handled. But we can find that after adopting these models, the degree of fit has been greatly improved.

5 Data Addition and Regression

5.1 Addition of exploratory data

In the above section (Exploratory Data Analysis), we found that the emissions of pm2.5 differ significantly between seasons or whether they are weekends or not. However, this information is not directly reflected in the original data set. But luckily, we can improve the effect of regression by mining the information of time to produce more features. Below are some of the features we produce from time.

Weekend: We think whether the impact on human activity (travel, factory work, etc.) is large on weekends, we think this will affect the pm2.5 content.

Season: We believe that the meteorological data in different seasons are very different, and the human behavior is also very different. We guess that the pm2.5 content of different seasons is also very different.

Time of the Day: We believe that natural factors such as temperature and wind speed vary greatly at different times of the day, and we guess that this will affect the content of pm2.5.

Holidays:We believe that holidays have a great impact on human activities. For example, May 1st and National Day are tourism booms, and during the Spring Festival, there are often limited travel behaviors. We think this will affect the content of pm2.5.

In addition to the first time, we also consider adding some specific time information, such as whether it is Monday morning, whether it is December evening, etc.

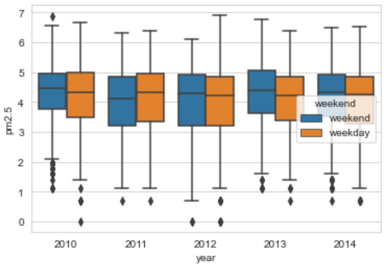
In the following figures, we draw some box plots of some factors, through which we can intuitively see that these factors do affect the content of pm2.5.

However, it is not rigorous to judge whether the added information has a significant impact on the dependent variable pm2.5 only based on the boxplots. So here we decided to take ANOVA. At the same time, in the EDA section, we can intuitively see that the distribution of pm2.5 is skewed, which obviously does not satisfy normality, so here we decided to adopt a non-parametric test method: *Kruskal-Wallis test* to check whether the added information has a significant impact on pm2.5. The result is given below:

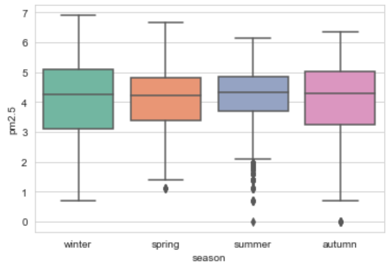
Table 6: Kruskal-Wallis test for the added features

Added information	**KW chi-squared**	**p-value**
weekend	27.643	1.459e-07
season	67.758	1.289e-14
daytime	403.56	< 2.2e-16
Nation_day	0.48721	0.4852
spring_festival	0.026246	0.8713
Monday_morning	9.9426	0.001615
Deceweekend_evening	27.673	1.437e-07
policy	3.1315	0.07679

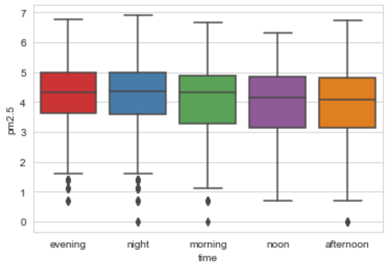
We found that in addition to the features 'Nation_day' and 'spring_festival', the other added information has a significant impact on the content



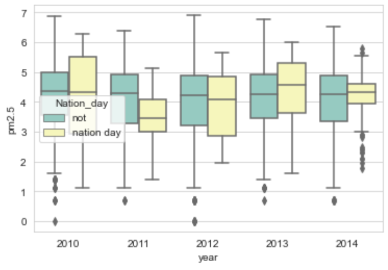
(a) Comparsion of PM2.5 between weekend and weekday in five years.



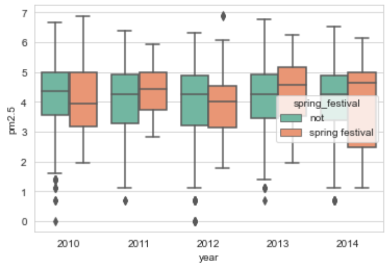
(b) Comparison of PM2.5 in four different seasons.



(c) Comparison of PM2.5 in five time periods of the day.



(d) Comparsion of PM2.5 during the national day or not.



(e) Comparsion of PM2.5 during the Spring Festival or not.

Figure 18: Comparsion of PM2.5 between weekend and weekday in five years.

of pm2.5, but at the same time we can see according to (c) and (d) of figure18, 'Nation_day' and 'spring_festival' seems to be obvious to the content of pm2.5 between different years.

In general, we can think adding these information can improve performance of our model and make our predictions more accurate.

5.2 Regression

In this part, we will repeat the regression models in section 3, but the difference is that we have more features at this time and expect better prediction results. The results obtained are summarized in the table below: (Because the versions of bagging and adaboosting of linear and ridge regression do not perform well, they are not listed in the following tables.)

Table 7: Summary of different reression methods after data addition

Model	R^2 **(train)**	R^2 **(test)**	**MSE (test)**
Linear Model	0.495733	0.488355	1.259906
Ridge Model	0.495730	0.488412	1.259764
GBDT	0.646806	0.622201	0.930315
XGBoost	0.723265	0.634129	0.900943
LightGBM	0.693620	0.633809	0.901730

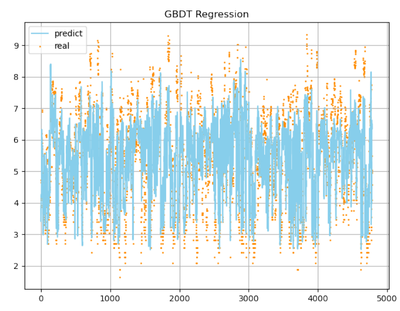
We can find that the goodness of fit obtained by the GBDT and two optimization algorithms based on GBDT (XGBoost and LightGBM) are still better than the linear models. So we only give the predicted figures obtained by these models here (figure 18).

From the above table, we can see that after adding this information, our model can indeed better fit and predict the content of pm2.5, (the best lightGBM can already reach 60% fit on the test set. (goodness of fit.)

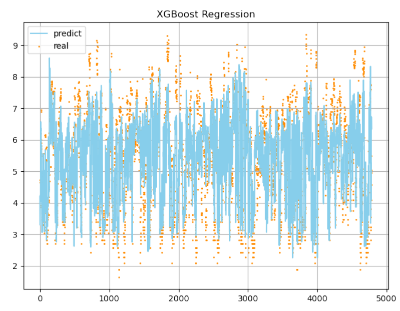
But we still have some aspects to improve: (1) Many of the features we use in this section have great correlation between them, and the importance of these features is not the same. We hope to obtain fewer but more important features to set our model (2) In the previous sections, we directly delete the observations with missing values, but these observations also contain lots of information of other features, we can fill in these missing values and make full use of all the information of the dataset to achieve better performance.

6 Data Filling and Regression

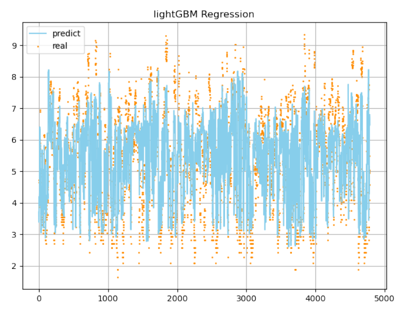
In the previous section, we adopted a strategy of direct deletion for observations with missing values, but such processing tends to lose lots of information, because these observations only contain one missing column 'pm2.5'. We can consider filling in missing values based on the data struc-



(a) Predicted figures of GBDT



(b) Predicted figures of XGBoost



(c) Predicted figures of LightGBM

Figure 19: Predicted figures of GBDT, XGBoost and LightGBM after the data addition.

ture of other non-missing features, thus making full use of all observations to improve the model's goodness of fit.

Here, we use two methods to fill in the missing values: Model-based kNN and Random Forest method.

6.1 Model-based kNN

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. For k-NN classification, an input is classified by a majority vote of its neighbors. That is, the algorithm obtains the class membership of its k neighbors and outputs the class that represents a majority of the k neighbors.

Here we can treat the missing pm2.5 values as Y, other explanatory variables as X ; take the data without missing values as our training set to train a regression KNN model. We can take the data with missing values as our test set to predict the miss values by the trained KNN model after training.

6.2 Random Forest

In this subsection, we use the Random Forest to predict the missing values. Random forest is a classifier that consists of multiple decision trees, and its output category is determined by the mode of the categories output by the individual trees. KNN and Random Forest share some similarities in some sense. It turns out that both algorithms can be viewed as so-called "weighted neighbor schemes". These are in the dataset $\{(x_i, y_i)\}_{i=1}^n$ computes a prediction $\hat{y}$ for a new point x by looking at the point's neighbors, and these neighbors are weighted using the weight function W. By apply this function W, we can fill the missing data with a high accuracy.

6.3 Results of Regression

We used the complete data set obtained based on the above two filling methods to perform different methods of fitting, and the results obtained are shown in the following tables:

For the dataset which applies the model-based kNN to fill in the missing values:

Table 8: Comparsion of different regression methods after data filling (Model-based kNN).

Model	R^2 **(train)**	R^2 **(test)**	**MSE (test)**
Linear Model	0.481052	0.469267	1.358172
Ridge Model	0.481049	0.469393	1.357850
GBDT	0.650555	0.615671	0.911259
XGBoost	0.726360	0.629119	0.879373
LightGBM	0.684319	0.629810	0.877736

Comparing the results of data without filling the missing data, we find that there is a slight decrease in the goodness of fit for both the linear models and the decision tree-based boosting algorithms, which shows that the model we use for filling the missing data may not satisfy the structure of the data, but hinder the fitting process.

For the dataset which applies the random forest to fill in the missing values:

Table 9: Comparsion of different regression methods after data filling (Random Forest).

Model	R^2 (train)	R^2 (test)	MSE (test)
Linear Model	0.501761	0.490602	1.297538
Ridge Model	0.501758	0.490669	1.297367
GBDT	0.652647	0.624229	0.957163
XGBoost	0.725343	0.637425	0.923552
LightGBM	0.704088	0.639624	0.917949

Different from the model-based kNN, we find that there is a slight increase in the goodness of fit for both the linear models and the decision tree-based boosting algorithms compared the results of data without filling the missing data, which shows that filling missing values through random forest is effective to a certain extent.

So far, our best fitted model based on removing 5% points of the original data can achieve a goodness of fit to nearly 64% on the test set.

7 Feature importance

Feature importance, we generally use to observe the contribution of different features. We naturally think the variable with high rank it's important.

This idea is usually used for feature selection. Remove the tail features with low contribution not only enhances the robustness of the model, but also plays the role of feature dimension reduction at the same time.

Another aspect is the interpretability of the model. Our desired outcome is that important features are intuitive and features that are intuitive are ranked high.

In this section, we mainly analyze the feature importance of the tree model that performs well in regression tasks, XGBoost. Here we compare the importance of each variable through two feature importance calculation methods.

7.1 Weight-based evaluation

Here the word 'weight' refers to the number of times a feature is used to split the data across all trees.

The weight counts the number of times a feature is used in all trees. As we know, the features in the decision tree of the base classifier can be reused in the XGBoost model. When a feature is continuously selected as a split node , we can think this feature is more important than other features.

Here, we use the function *plot_importance* to draw all feature importances, the figure is given below:

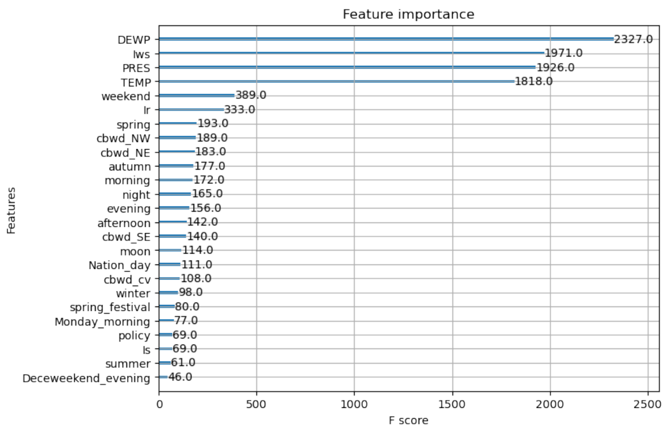


Figure 20: Feature importance of XGBoost in F score

From the above figure, we can find that the four most important features based on *weight* are the features of the original data: 'DEWP', 'Iws', 'PRES' and 'TEMP'. In addition, in the data we added, the more important features are 'weekend' and 'spring'. They represent whether the observation is on the weekend and whether it is in the spring, respectively.

7.2 Gain-based evaluation

In addition, we also adopt a gain-based feature importance evaluation method.

Here the word 'gain' refers to the average gain across all splits the feature is used in. Gain is a generalized concept of information gain, which means that when the node is split, the feature brings the average value of optimization of the information gain (objective function).

Here we want to apply the function *feature_importances_* to compute the gain of different features.

Among all the feature importances based on gain, the gain of 'winter' is the highest and is much more important than the rest of the features, which is followed by 'cbwd_SE', 'summer', and 'cbwd_NW', where only the feature 'summer' is added.

The added features 'winter' and 'summer' represent whether an observation is in winter or summer, respectively.

Table 10: The Gain and the ranks of gain of different variables

Variable	gain	Rank
DEWP	0.07310177	5
TEMP	0.03856876	8
PRES	0.01167355	15
Iws	0.04058880	6
ls	0.00523531	22
lr	0.01500035	11
cbwd_NE	0.02466225	9
cbwd_NWM	0.08447220	4
cbwd_SE	0.14765884	2
cbwd_cv	0.00829200	20
weekend	0.00773573	21
spring	0.03932472	7
summer	0.09507837	3
autumn	0.01439243	12
winter	0.30055340	1
morning	0.00953915	17
moon	0.00840928	19
afternoon	0.00489780	23
evening	0.01368284	13
night	0.01301933	14
spring_festival	0.01016174	16
Nation_day	0.01633548	10
Deceweekend_evening	0.00426607	24
Monday_morning	0.00399283	25
policy	0.00935704	18

8 Classification

8.1 Introduction of classification

In the above regression model, we can already achieve a relatively good prediction effect for the PM2.5 content, but in most cases, we do not care about the specific PM2.5 content in each hour, we prefer to know what level of air pollution is in a certain period of time and whether it is suitable to go out.

By applying the classification of pollution degree of Section 2.3, we can easily identify the pollution level at this time. When it is still lightly polluted(pm$\leq$ 35), we can consider going out, but not when it is moderately(35<PM$\leq$150) or heavily polluted(PM>150). It maybe be a wiser choice to stay home.

Therefore, it is necessary to classify the pollution degree according to the content of pm2.5, and to train a training model on these processed data. In this section we use a considerable number of classification methods and compare the accuracy obtained by these methods.

8.2 Logistic Regression

Logistic regression is a common classification method, it is similar to linear classification, but it use sigmoid function as the discriminant function and then the maximum likelihood estimation is used to update the parameters of the classifier so that the effect of the classifier is constantly evolving. The result is as following :

```
Accuracy in train set: 0.689136
Accuracy in test set: 0.701854
```

8.3 Artificial Neural Network

ANN refers to a complex network structure formed by interconnecting a large number of processing units (neurons), and is a kind of abstraction, simplification and simulation of the structure and operation mechanism of the human brain. Artificial Neural Network (ANN), which simulates neuron activity with a mathematical model, is an information processing system established based on imitating the structure and function of the neural network of the brain.

In this part, we apply neural network to the classification of pm2.5 data. We use temperature, humidity, wind direction, wind speed and so on as dependent variables. At the same time, in order to prevent additional influencing factors caused by different orders of magnitude between dependent variables, we normalized each dependent variable to ensure that different dependent variables are in the same order of magnitude, so as to achieve better training effects. We use teh hidden_layer = (20, 10), relu activate function, adam optimizer to train 5000 epoch, get the result as following :

```
Accuracy in train set: 0.781756
Accuracy in test set: 0.716393
```

8.4 Decision Tree

A decision tree is a type of supervised machine learning used to categorize or make prediction based on how the previous set of questions were answered.

The classification tree predicts the classification result based on one or more input variables and combined with the division conditions. The splitting process starts from the root node of the classification tree: at each node, the algorithm will check whether the input variable needs to be divided intermittently to the left cotyledon and the right cotyledon according to the division conditions. Stop splitting when a child node (end point) of any classification tree is reached. The accuaray of classification is as following:

```
Accuracy in train set: 0.6125385
Accuracy in test set: 0.6124167
```

8.5 K-Nearest Neighbor

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. For k-NN classification, an input is classified by a majority vote of its neighbors. That is, the algorithm obtains the class membership of its k neighbors and outputs the class that represents a majority of the k neighbors. In this paper, we can treat pm2.5 values as y , other variables as x ; take the data without missing values as our training set to train a classification or regression KNN model, after training, we can take the data with missing values as our test set to predict the miss values by the trained KNN model. The accuaray of classification is as following:

```
Accuracy in train set: 0.814321
Accuracy in test set: 0.650012
```

8.6 Support Vector Machine

SVM is a kind of generalized linear classification that performs binary classification on data according to supervised learning, and its decision boundary is the maximum margin hyperplane that solves the learning samples. Given a set of training instances, each marked belonging to one or the other class, the SVM training algorithm builds a model that assigns new instances to one of the two classes, making it a non-probabilistic two Binary Linear Classifier. The SVM model is to represent instances as points in space such that the mapping makes instances of different classes separated by as wide a noticeable interval as possible. Then, map the new instances to the space and predict the class they belong to based on which side of the interval they fall on. The accuaray of classification is as following:

```
Accuracy in train set: 0.7537859
Accuracy in test set: 0.7025934
```

8.7 Random Forest

In machine learning, a random forest is a classifier that consists of multiple decision trees, and its output category is determined by the mode of the categories output by the individual trees. Lin and Jeon in 2002 pointed out the relationship between the random forest algorithm and the K-nearest neighbor algorithm (k-NN). It turns out that both algorithms can be viewed as so-called "weighted neighbor schemes". These are in the dataset $\{(x_i, y_i)\}_{i=1}^n$ computes a prediction $\hat{y}$ for a new point x by looking at the point's neighbors, and these neighbors are weighted using the weight function W. The accuaray of classification is as following:

```
Accuracy in train set: 0.835218
Accuracy in test set: 0.729241
```

8.8 Summary

In this part, in order to more easily determine whether we can go out by judging the level of pollution, that is, to solve the classification problem of environmental pollution levels, we have tried many different classification algorithms, and their classification accuracy on the test set are not exactly equal.

The accuracy of different classification algorithms are summarized in the table below:

Table 11: Summary of different classification methods.

Model	Accuracy (train)	Accuracy (test)
Logistic	0.689136	0.701854
ANN	0.781756	0.716393
DT	0.612539	0.612420
kNN	0.814321	0.650012
SVM	0.753786	0.702593
Random Forest	0.835218	0.729241

It can be seen that whether it is on the training set or the test set, using random forest for classification is a better choice. On the test set, the accuracy rate of random forest classification can reach close to 73%.

9 Time Series Analysis

In fact, in addition to the regression and classification perspective, we can also analyze this data from another perspective. Aside from all other explanatory variables and only observe the response variable pm2.5, we can find that this is a very obvious time series, so we can also try to predict this data through time series analysis.

9.1 Data Processing for Time Series Analysis

From the data, it can be found that there are missing values in the pm2.5 data column. The mean of Beijing pm2.5 from 2010 to 2014 is 98.6, the median is 92.0, and the interval is [0, 994].

The pm2.5 data reflects the pm2.5 value at a certain time of a certain day. Observing the missing values, it is found that some values are missing on a certain day, and some are missing values at certain moments of a certain day. It is planned to count the value of pm2.5 in units of days, so if there is a missing value in a certain day, delete the pm2.5 value at all times of the day, that is, the method to deal with the missing value is to delete the observation.

For observations with no missing values for 24 hours in a day, we take the average of all the data of that day to represent the PM2.5 content of that day.

After data cleaning, data integration and data transformation, the final available data quantity is as follows, the unit is: days.

```
2010:  320
2011:  288
2012:  300
2013:  330
2014:  333
```

Next we will use the pm2.5 data of these days for time series analysis.

9.2 Time Series Analysis Modeling

Analyze the pm2.5 value in the past five years in units of days, and draw a line plot to observe his general trend.

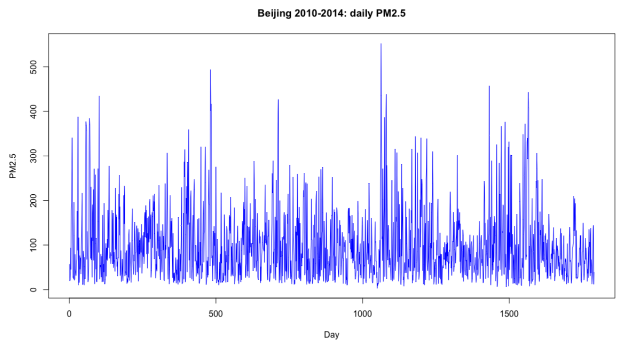


Figure 21: Daily PM2.5 contents in Beijing from 2010 to 2014.

The requirement of ARIMA model for time series is to be stationary. From the above figure, we can find that there is no fixed upward or downward trend. A rough judgment is that it is a stationary sequence. No difference operation is performed, and the ADF unit root stationarity test is used to test the stationarity of the series.

```
Augmented Dickey-Fuller
Testdata:  data$pm2.5
Dickey-Fuller = -9.8766, Lag order = 12,
  p-value = 2.368039e-30
alternative hypothesis: stationary
```

The null hypothesis of the ADF test is that there is a unit root. As long as the statistical value is less than the number at the 1% level, the null hypothesis can be rejected very significantly, and the data of pm2.5 is considered stationary.

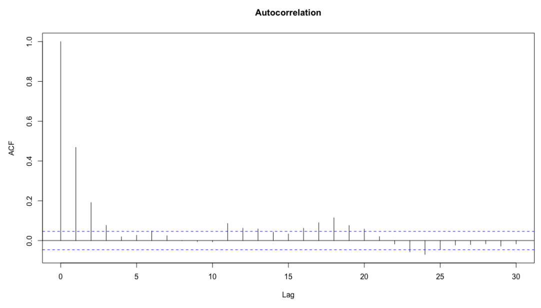
Table 12: General Behavior of ACF and PACF for ARMA Models

Stationary Models	ACF	PACF
AR(p)	Tails off	Cuts off after lag *p*
MA(q)	Cuts off after lag *q*	Tails off
ARMA(p,q)	Tails off	Tails off

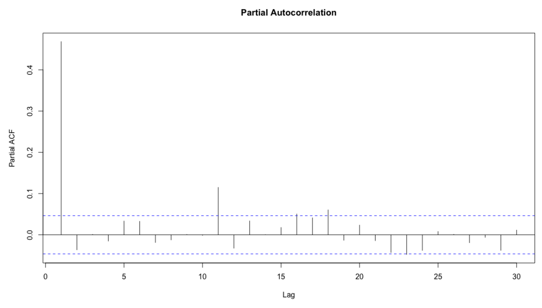
Observing the images of ACF and PACF, we found that the ACF image has 3 orders outside the confidence interval, and the PACF image has 1 order outside the confidence interval, so we consider three models: MA(3), AR(1), ARMA(1,3).

Table 13: General Behavior of ACF and PACF for ARMA Models

Models	estimated sigma2	log likelihood	AIC
AR(1)	4691	-10100.27	20206.93
MA(3)	4682	-10098.46	20206.53
ARMA(1,3)	4682	-10098.42	20208.84



(a) Autocorrelation function of pm2.5 in Beijing



(b) Partial Autocorrelation function of pm2.5 in Beijing

Figure 22: ACF and PACF of pm2.5 in Beijing.

By predicting the estimated σ^2 and AIC, we finally choose MA(3) as our prediction model. Next, we will perform model diagnostics on this model.

9.3 Model Diagnostics

Model diagnostics, or model criticism, is concerned with testing the goodness of fit of a model and, if the fit is poor, suggesting appropriate modifications.

Here, we consider taking the residual analysis. If the model is correctly specified and the parameter estimates are reasonably close to the true values, then the residuals should have nearly the properties of white noise: (1) independent (2) identically distributed normal variables (3) zero means (4) common standard deviations.

Plots of Residuals

Our first diagnostic check is to inspect a plot of the residuals over time. If the model is adequate, we expect the plot to suggest a rectangular scatter around a zero horizontal level with no trends whatsoever.

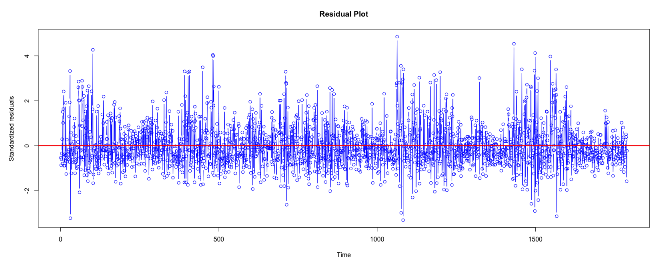


Figure 23: Standard residuals of the pm2.5 sequential MA(3) model

Overall, the time series plot of residuals is around the zero horizontal line and has no obvious trend. But it can be clearly seen that there are a few points in the sequence with magnitudes of residuals larger than 3—which is very unusual in a standard normal distribution.

Normality of the Residuals

Quantile-Quantile plots are an effective tool for assessing nor- mality. Here we apply them to residuals and the Q-Q plot are given below:

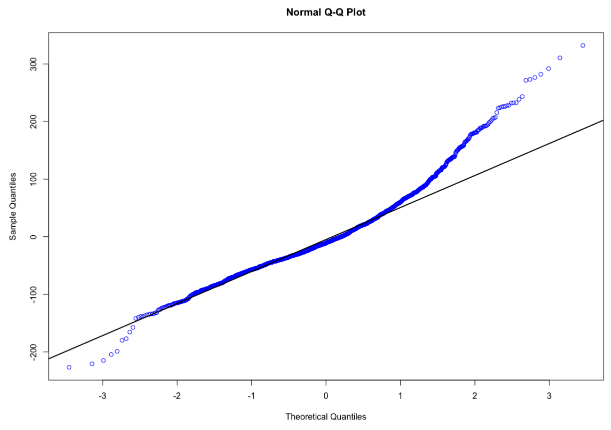


Figure 24: Quantile-Quantile plot of pm2.5 sequence MA(3) model residuals

This figure shows a quantile-quantile plot of the residuals from the MA(3) model estimated for the pm2.5 series. It can be seen that these points deviate significantly from the straight line at large positions, which makes us have to doubt the correctness of the normality assumption. In addition, the *Shapiro-Wilk normality test* applied to the residuals produces a test statistic of W=0.94958, which corresponds to a p-value less than 0.01, and we would reject normality based on this test. The reulst is given below:

```
Shapiro-Wilk normality testdata:
residuals(model2)
W = 0.94958, p-value < 2.2e-16
```

Autocorrelation of the Residuals

To check on the independence of the noise terms in the model, we consider the sample autocorrelation function of the residuals. A graph of the sample ACF of these residuals are shown in the following figure. The dashed horizontal lines plotted are based on the large lag standard error of $\pm\frac{2}{\sqrt{n}}$. There is no evidence of autocorrelation in the residual of this model.

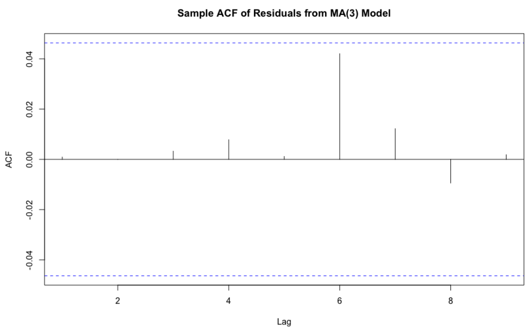


Figure 25: Sample ACF of residuals of pm2.5 sequence MA(3) model.

We conclude that this figure does not show statistically significant evidence of nonzero autocorrelation in the residuals.

Ljung-Box Test

In addition to looking at residual correlations at individual lags, it is useful to have a test that takes into account their magnitudes as a group. For example, it may be that most of the residual autocorrelations are moderate, some even close to their critical values, but, taken together, they seem excessive.

Here, we use the function *tsdiag()* to perform the Ljung-Box test for different lag values. The results are given below:

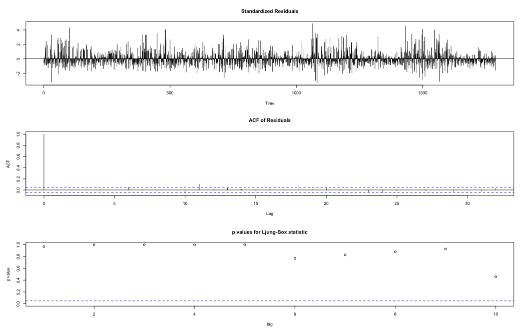


Figure 26: Diagnostic display of the pm2.5 sequence MA(3) model.

Figure 26 shows three of our diagnostic tools in one display — a sequence plot of the standardized residuals, the sample ACF of the residuals, and p-values for the Ljung-Box test statistic for a whole range of values of K from 1 to 10. The horizontal dashed line at 5% helps judge the size of the p-values. In this instance, everything looks very good. The estimated MA(3) model seems to be capturing the dependence structure of the pm2.5 property time series quite well.

9.4 Model Predictions

In the above section, we selected the most suitable forecasting model MA(3) based on estimated σ^2 and AIC criteria. Diagnosed this model and found that it captures the dependence structure of the pm2.5 property time series quite well. The only shortcoming is that the residuals of the model MA(3) do not conform to the normal distribution, but in general we think that our time series model is reasonable.

Next, we will show the comparsion between our model's predictions and real data.

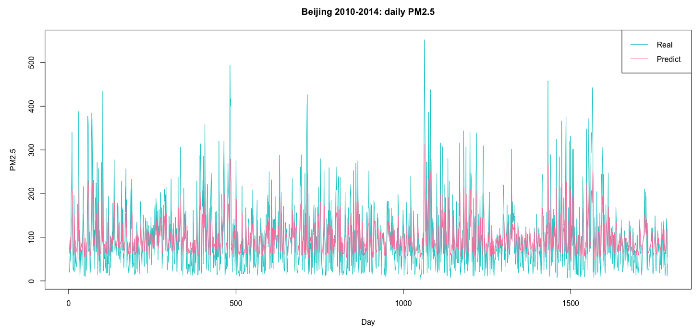


Figure 27: The predicted figure of the pm2.5 sequence MA(3) model.

The figure is a visualization of real data and model fitted data, where the blue line is the visualization of the original data, and the pink line is the visualization of the model's prediction on the blue data. As can be seen from the figure, the model basically simulates the trend of the original series.

10 Conclusion

In this paper, we analyze pm2.5 data from three major aspects: regression, classification and time series analysis. In the first part, we applied the several regression methods to analyse the pm2.5 data with some additional features, and get the best result about $R^2 = 0.723265$ in training set and $R^2 = 0.634129$ in test set by using XGBoost method. Then, we apply the data filling algorithm to the missing data, then trained with these complete datas, get the result about $R^2 = 0.704088$ in training set and $R^2 = 0.639624$ by using LightGBM method and Random Forest filling algorithm which is a little better than before filling in the missing values. In the second part, to help people have a more intuitive understanding of pm2.5 content, we divided the data into three levels according to pm2.5 content: low, polluted and high. After doing that, this problem becomes an obvious classification task, and we applied several classification method to classify it, and get the best result about accuracy = 0.835218 in training set and accuracy = 0.729241 in test set by using Random Forest method. In the last part, we applied time series analysis to just analyse the pm2.5 content, which turns the original problem into a time-dependent problem. We applied stationary MA(3) model to analyze it, basically meet the requirements of model diagnosis. And in general, it has a good predictive effect.