

Title: find influence factors and predict current value/price of used cars

Author: Hangliang Ren

1. **Summary of research questions and Results:**

- a) Which are top 10 (or maybe 15) most important influence factors to the current price of used cars?

Answer: These are top 10 most important / relevant influence factors: ['regDate', 'model', 'bodyType', 'creatDate', 'v_0', 'v_2', 'v_3', 'v_5', 'v_11', 'v_12'].

- b) Use three different ML methods:

- linear regression (learned in class)
- ridge regression (similar to least squares method)
- random forest (not only can do classification, interesting)

with features chosen in a) to predict what the current price is of each used car. Find which method is better?

Answer: Random forest method is better.

- c) Build a neural network to predict what the current price is of each used car. Find which one, our chosen traditional ML methods, or neural networks, is better in our current project?

Answer: Random forest method (our chosen traditional ML methods) is better in our current project.

2. **Motivation and background:**

- a) This project is inspired by a kaggle project, link:
<https://www.kaggle.com/c/1056lab-used-cars-price-prediction/overview>
- b) Since recent years, with the trend leading by Tesla, it has become very popular to use electric cars to replace fuel cars. At the same time, after being replaced, those fuel cars will not disappear directly; instead, a huge proportion of those replaced cars will be sold again in the second hand market. Because the sales amount of electric cars is increasing so fast, it is expected that the number of second hand cars for sale will also increase very fast during the future 5-10 years. However, one of the main problems, which people face when buying second hand cars now, is that the given price by sellers is normally higher than the actual market value. In many occasions, after buying a second hand car, customers can often find a more fair price of the same type car provided by another seller. Thus, it is very meaningful to develop a **model** to evaluate a **relatively accurate market value/price** of second hand cars, providing this estimated price to customers as a reference, helping them make wiser decisions when buying second hand cars.
- c) Also, nowadays a huge number of mass media, tech companies, and experts are all publicizing that Machine Learning & Deep Learning are becoming more and more important to influence our daily lives. Personally, I feel very interested in this, so I want to take opportunities like this project to check by myself whether Machine Learning & Deep Learning are so cool and magical as those people publicizing. Besides this, I also want to take this project to have a glance at which one, ML or Deep Learning, is more accurate in prediction in this field (price estimation).

3. **Dataset:** Just as mentioned above, I actually do not have access to the dataset of that kaggle project. As an alternative, I will use an open-sourced dataset (not related to the original project) from Alibaba Cloud. One thing to note is that accessing the dataset on Alibaba Cloud is inconvenient (must use phone number to register), so I store it directly in the google drive.

Here's the link (original dataset **has not been done by any preprocess, such as handling missing data**):

<https://drive.google.com/file/d/1fgUXVEAmnvWJVNXFKqOrgHTLkgG66Hoj/view?usp=sharing>

Note: If you want to use pandas to check this dataset by yourself, do note that columns are not separated through commas, but through whitespace. (need some more parameters when calling read_csv)

Details about this dataset:

- a) In part 1, I mentioned that this dataset is better than the original dataset, because it has 150000 rows/samples, 31 columns/attributes vs. 6019 rows/samples, 13 attributes in the kaggle dataset. With this much huger dataset, it is very helpful to realize those courageous goals set in part 2.
- b) There are attributes in our dataset:
['SaleID', 'name', 'regDate', 'model', 'brand', 'bodyType', 'fuelType', 'gearbox', 'power', 'kilometer', 'notRepairedDamage', 'regionCode', 'seller', 'offerType', 'creatDate', 'price', 'v_0', 'v_1', 'v_2', 'v_3', 'v_4', 'v_5', 'v_6', 'v_7', 'v_8', 'v_9', 'v_10', 'v_11', 'v_12', 'v_13', 'v_14']
Among them, **v_0 to v_14 are anonymous variables.**
- c) The whole dataset has information about 150000 used cars, I will use 100000 for training set, 50000 for test set.

4. **Methodology:**

a) **Data cleaning & Feature engineering:**

- (1) Clean the data by filling missing values. This time, to fill missing values, we will use **Lagrange Polynomial (update: we now use normal polynomial interpolation with high order)**. For more information about how it works mathematically, check the link below.
https://en.wikipedia.org/wiki/Lagrange_polynomial
- (2) Separate dataset into training set (100000 samples) and test set (50000 samples). Package chosen factors as features, price as the label.
- (3) To find the top 10 most important influence factors to the current price of used cars, we will do the following steps. ①Use **heatmap** to identify top 10 attributes most correlated/relevant to the current price. **Include the graph and our observation in the report.** ②Use **pearson correlation coefficient** to compare each individual attribute/column with the price attribute/column, double checking whether the correlation exists. Generally, if this coefficient ≥ 0.8 , then we can say the correlation exists. **Discuss (in testing part) our observation and possible doubt (e.g., computed result < 0.8) in the report.** ③Choose 3 attributes/factors from those 10 factors, using **trendline** to draw graphs of each attribute vs. price, observing the correlation (e.g., when x value increases, y value increases). **Include graphs and discuss our observation in the report.**

b) **Machine Learning (** Challenge goal, Machine Learning):**

- (1) Build a linear regression model, train it through features chosen above (those 10 factors), and use it to predict the current price of each used car. Calculate the mean squared error and plot the learning curve. **Store the mean squared error & learning curve in report.**
- (2) Build a ridge regression model, train it through features chosen above (those 10 factors), and use it to predict the current price of each used car. Calculate the mean squared error and plot the learning curve. **Store the mean squared error & learning curve in report.**
- (3) Build a random forest model, train it through features chosen above (those 10 factors), and use it to predict the current price of each used car. Calculate the mean squared error and plot the learning curve. **Store the mean squared error & learning curve in report.**
- (4) Compare calculated mean squared errors from three models, **discuss observation in report.** Observe and compare learning curves of three models, **discuss result in report (e.g., overfit or underfit, whether result blows up, and so on).** Combine observation and comparison results above, briefly **talk about which model works better** in our price prediction project (in the accuracy/error scale).

c) **Deep Learning (** Challenge goal, New Library tensorflow):**

- (1) Build a neural network with two hidden layers, use **sigmoid** as activation function, use **L2 normalization** in the loss function, and set learning rate between 0.1-0.2, training the model. Calculate the mean

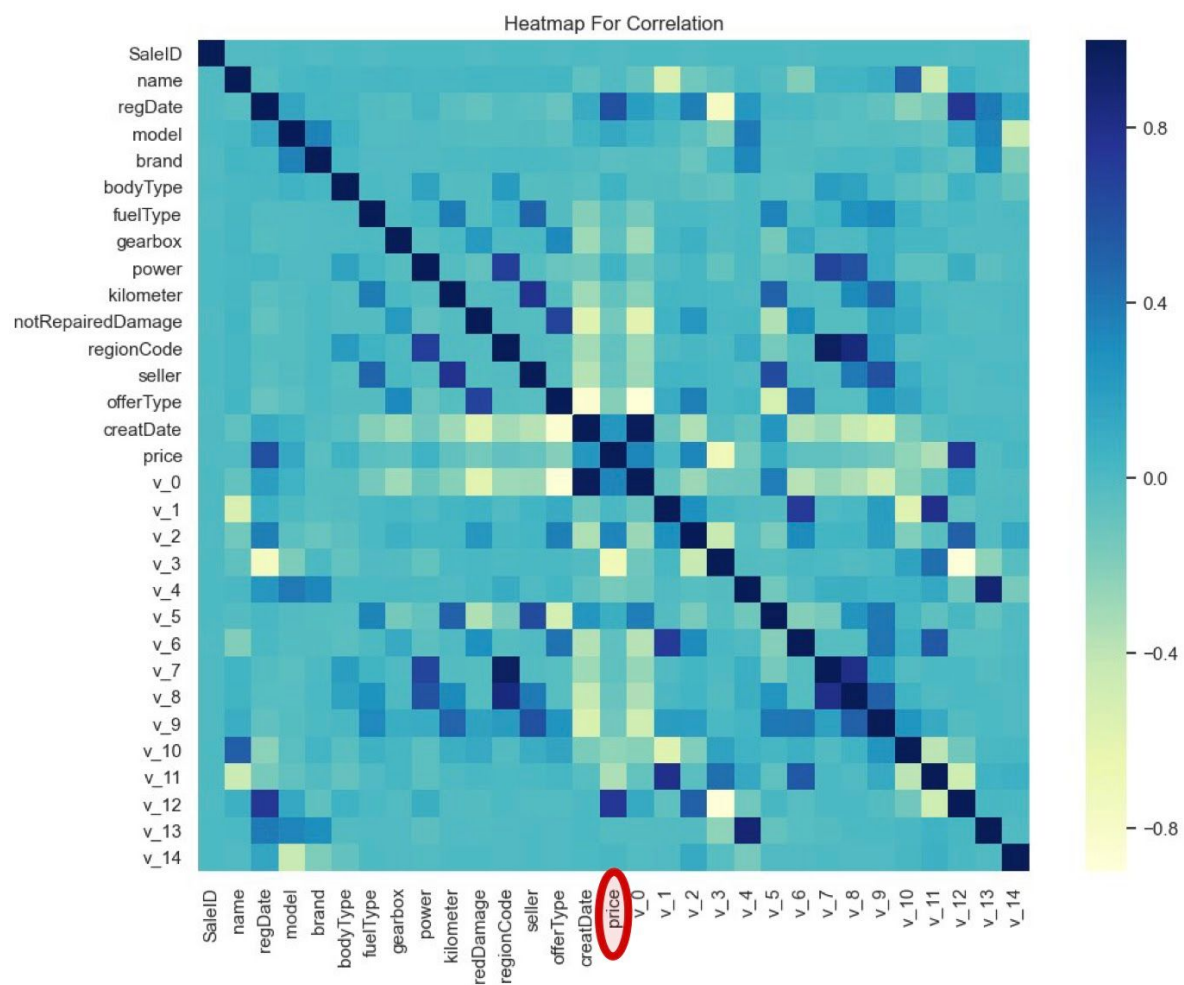
squared error and plot the learning curve. **Store the mean squared error & learning curve in report.**

- (2) Build a neural network with two hidden layers, use **relu** as activation function, use **L1 normalization** in the loss function, and set learning rate between 0.1-0.2, training the model. Calculate the mean squared error and plot the learning curve. **Store the mean squared error & learning curve in report.**
 - (3) Build a neural network with two hidden layers, use **relu** as activation function, use **L2 normalization** in the loss function, and set learning rate between 0.1-0.2, training the model. Calculate the mean squared error and plot the learning curve. **Store the mean squared error & learning curve in report.**
 - (4) Compare calculated mean squared errors from three models, **discuss observation in report (update: we now discuss more about why our Deep Learning models do not perform well and possible solutions).** Observe and compare learning curves of three models, **discuss result in report (e.g., overfit or underfit, loss, accuracy, and so on).** Combine observation and comparison results above, briefly **talk about which model works better** in our price prediction project (in the accuracy/error scale).
- d) **Final comparison & Result:** Choose the model trained by traditional Machine Learning methods in part b), with the smallest value of loss. Choose the model trained by Deep Learning methods in part c), with the smallest value of loss. Compare these two models regarding mean squared errors, loss, learning curves, etc. **In report, discuss our comparison result, and which model works better** for our price prediction project.

5. Results:

- a) **Question:** Which are top 10 (or maybe 15) most important influence factors to the current price of used cars?

Answer: Based on the description in methodology, after performing polynomial interpolation filling missing values, we then begin to look for top 10 most important / relevant influence factors to the current price of used cars. To do this, we check the correlation between each attribute and the current price, if the correlation for a specific attribute is higher, then it means that this attribute is a more relevant influence factor to the current price. Therefore, we use heatmap to visualize the correlation value for all attributes, focusing on the column for price.



Here is the plotted heatmap. (with heatmap_analysis method in data_process.py)

The circled column is the column for price. In this column, if a square is **very dark** (blue) in color, then it means that the attribute represented by the related row has strong positive relationship / correlation with price attribute; if a square is **very light** in color, then it means that the attribute represented by the related row has strong negative relationship / correlation with price attribute. Therefore, we want to select attributes represented by squares whose color is the very dark or very light, both kinds of color means very strong correlation with price attribute.

Thus, through high darkness / lightness in color, we select our top 10 most important / relevant influence factors: ['regDate', 'model', 'bodyType', 'creatDate', 'v_0', 'v_2', 'v_3', 'v_5', 'v_11', 'v_12'].

Note: Tests about correlation between these attributes and price will be in Testing part.

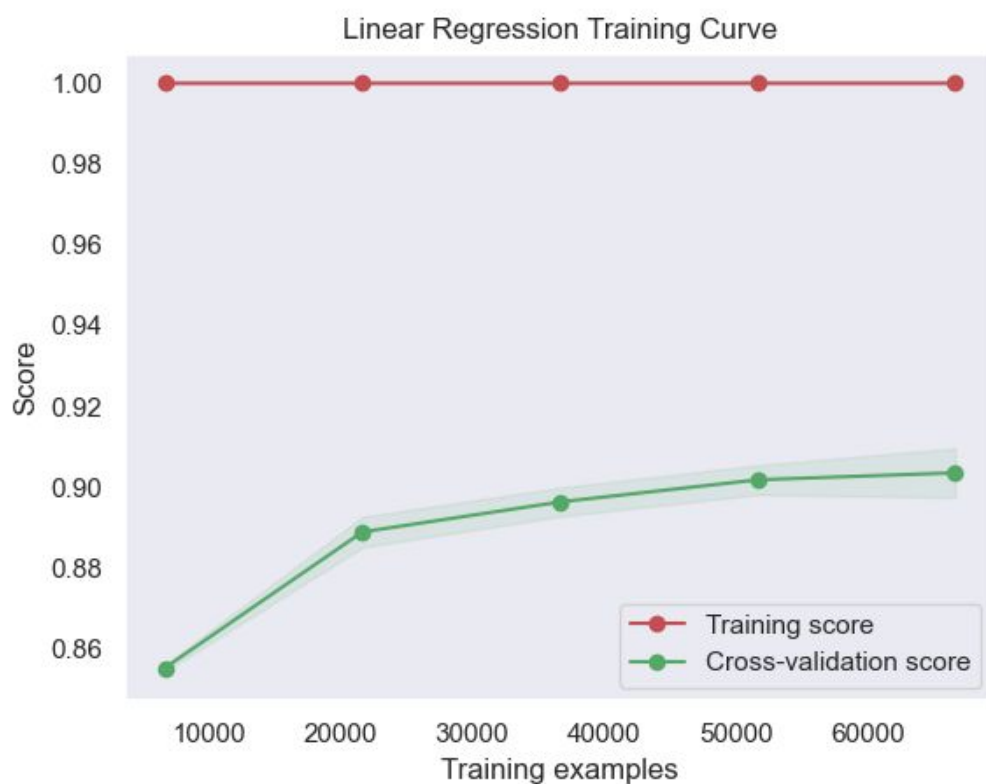
b) **Question:** Use three different ML methods:

- linear regression (learned in class)
- ridge regression (similar to least squares method)
- random forest (not only can do classification, interesting)

with features chosen in a) to predict what the current price is of each used car? Find which method is better?

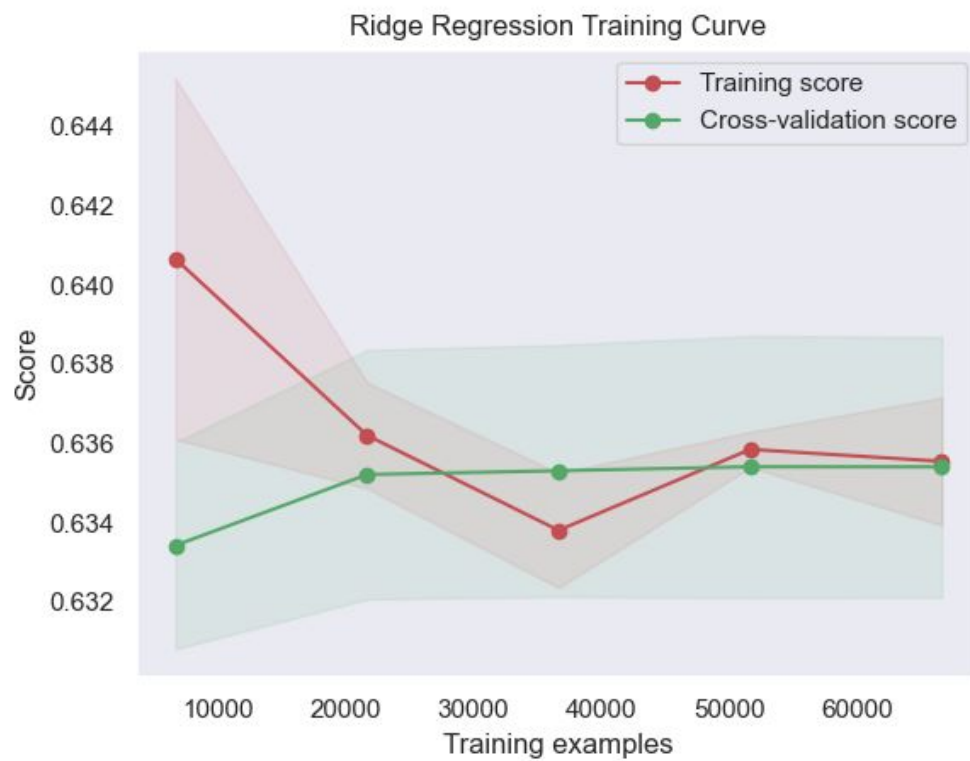
Answer: Through building models with linear regression, ridge regression, and random forest, and training these three models, we get the learning curve and mean squared errors for them. Here's the plots and errors. (**with plot_learning_curve method in machine_learning.py**)

learning curve for linear regression model:



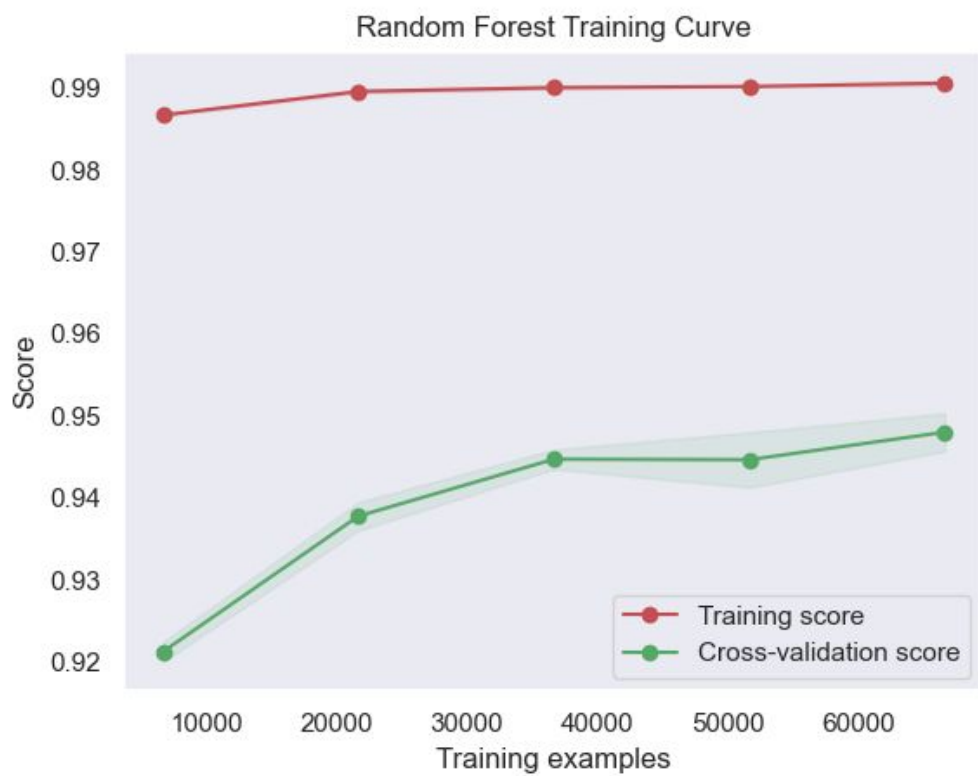
error: 94.37133519523694

learning curve for ridge regression model:



error: 413.9899166541826

learning curve for ridge regression model:



error: 56.24039307241358

Through observation, we can find that regarding **validation / test score, and learning curve**:

- 0.95 (random forest) > 0.90 (linear regression) > 0.636 (ridge regression)
- Overfit / Underfit do not occur in any of these three methods, because training score curves are generally above validation score curves.

and **mean squared error**:

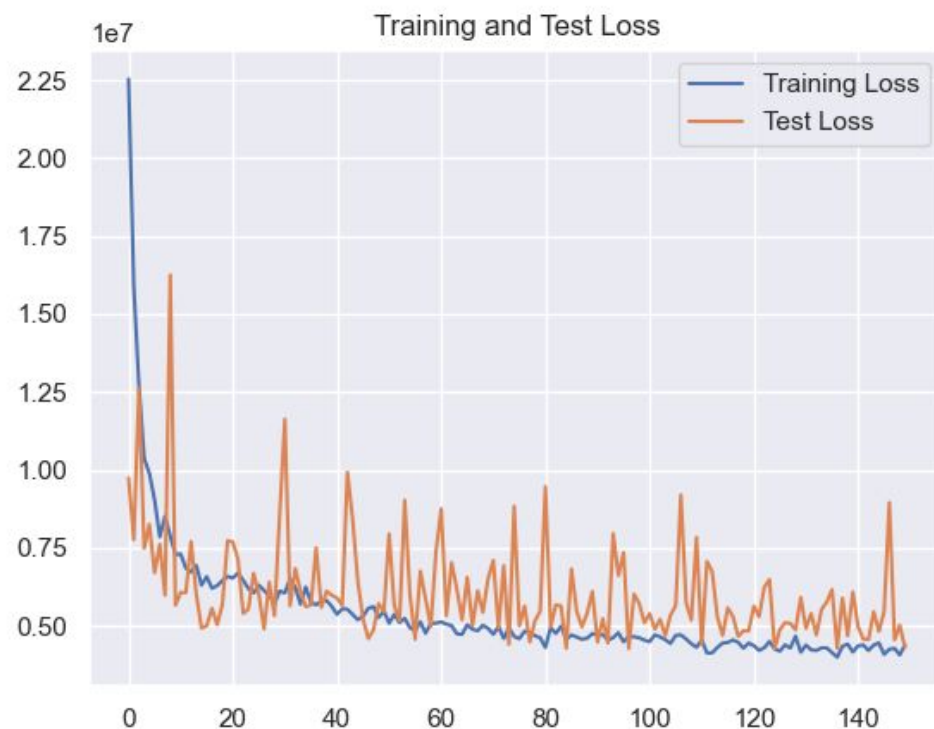
- 56 (random forest) < 94 (linear regression) < 414 (ridge regression)

model built & trained through **random forest** method has the best test score and the least error; thus, random forest method is better than linear regression / ridge regression.

- c) **Question**: Build a neural network to predict what the current price is of each used car? Find which one, our chosen traditional ML methods, or neural networks, is better in our current project?

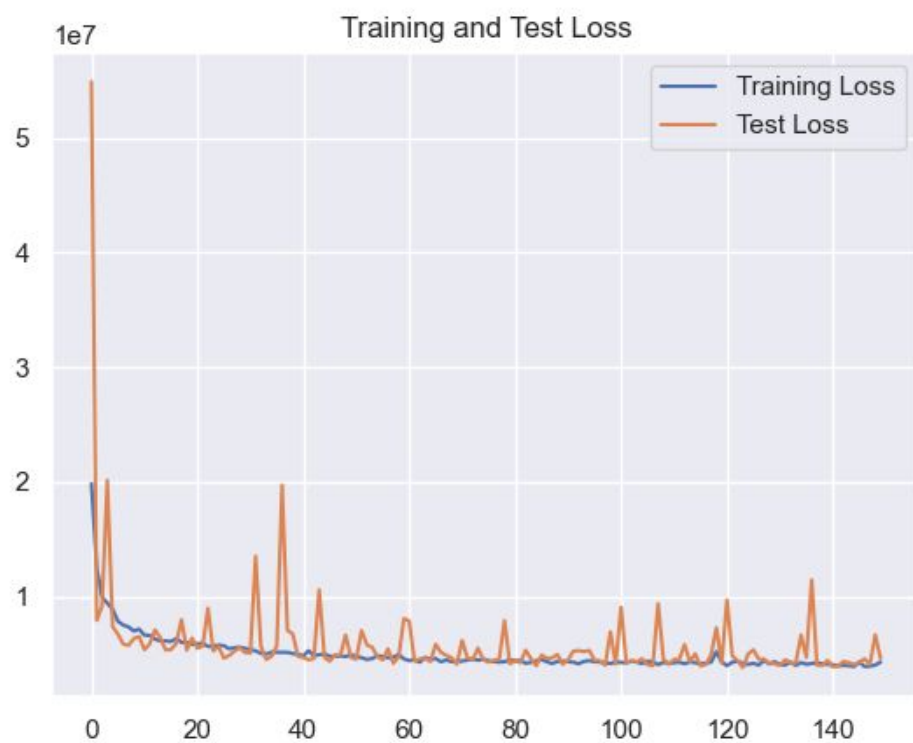
Answer: Through building models with neural network using l2 normalization & sigmoid activation, neural network using l1 normalization & relu activation, neural network using l2 normalization & relu activation, and training these three models, we get the learning curve and mean squared errors for them. Here's the plots and errors.

learning curve using neural_l2_sigmoid method:



error: 4300372.5

learning curve using neural_l1_relu method:



error: 4621339.5

learning curve using neural_l2_relu method:



error: 6789494.5

Through observing the learning curve and errors of these three models trained by neural networks, the performance of these models are not good / quite bad (error is up to 10^6 , very very high). Therefore, to the question “which one, our chosen traditional ML methods, or neural networks, is better in our current project”, the answer is obvious: in our object, any models trained by any traditional ML methods are better than every model trained by neural networks; specifically, **model trained by random forest (traditional ML method) is the best.**

At the same time, a more interesting topic / problem is to analyze why in our project, models trained by our designed neural networks perform so bad.

Analysis / Explanation: Based on reading materials, these are possible reasons for the low performance.

- Because the dataset is too huge, too fully “feed” the whole dataset into our neural network and finish the training, it will definitely need a much more powerful GPU than I have access to; thus, I choose to sample our dataset, feeding into the neural network with smaller version of dataset. However, even with some good sampling methods, our smaller version of dataset still cannot represent the whole original dataset, which makes our neural network cannot train the model considering all cases / samples / situations appeared in the dataset.
- Our original dataset is very huge (150000 samples), so a neural network with only two hidden layers is not enough to contain / analyze all information in the dataset; therefore we definitely need a much huger neural work with much more hidden layers & neurons.
- There exists time data in our dataset, but normal neural network is not good at dealing with data related to time.
- The situation our neural network facing now is underfit instead of overfit, but we still use many techniques dealing with overfit, such as normalization. Under our current situation, these techniques dealing with overfit will lower the performance of our network.

Possible Solutions:

- Under current situation, do not use techniques dealing with overfit such as normalization, because we are dealing with underfit instead of overfit.
- Use the complete dataset with a neural network much huge in scale (much more hidden layers & neurons). (of course, we need much more powerful GPU(s))
- Use LSTM neural network, which performs much better when dealing with data related to time.

6. **Challenge Goals:**

- a) **Machine Learning**. My project realizes the Machine Learning goal because in the project, I used three different ML methods, linear regression, ridge regression, and random forest to train models predicting the current price of used cars through data of chosen attributes. At the same time, I also use learning curves and mean squared errors to analyze which one of these ML methods performs better in our project.
- b) **New Library**. In my project, I realize the New Library goal through self studying tensorflow library, and implementing tensorflow to build three different neural networks training models. Admittedly, the final training performance is worse than those three traditional machine learning methods, because the mean squared errors of models trained by neural networks are too high. However, I further read more materials, combining knowledge I have with the situation / problem I am currently facing (too high errors), analyzing the cause of the problem, and proposing possible solutions.

7. **Work Plan Evaluation:**

a) **Data cleaning & Feature engineering (4 hours):**

- 2 hours for filling missing values, separate training & test datasets.
- 2 hours for observing data, finding 10 most correlated attributes, validating our findings, writing them into the report.

Final Feedback: Estimate of time is **accurate**, in practice, I spent around 4 hours and 15 minutes finishing this part's work.

b) **Machine Learning (7 hours):**

- 5 hours for building & training linear regression, ridge regression, and random forest models.
- 2 hours for deriving & observing & comparing mean squared errors and learning curves, and writing these findings into the report.

Final Feedback: Estimate of time is **accurate**, in practice, I spent around 4 hours and 30 minutes building & training machine learning models, and spend around 2 hours and 30 minutes analyzing & evaluating these trained models.

c) **Deep Learning (7 hours):**

- 5 hours for building & training three described (in part 5) neural networks.
- 2 hours for deriving & observing & comparing mean squared errors and learning curves, and writing these findings into the report.

Final Feedback: Estimate of time is **a bit inaccurate**, in practice, I spent around 5 hours building neural networks & training models; however, after I finished training and began analyzing, the mean squared error was way much higher than I expected. Thus, I spent more time than I expected, around 6 hours, reading more about related materials & documents, analyzing models & possible reasons causing such low performance in models trained by neural networks, and proposing possible solutions for such low performance.

d) **Final comparison & Result (2 hours):**

- 2 hours for choosing the best Machine Learning model and the best Deep Learning model, observing & comparing mean squared errors and learning curves, and writing findings into the report.

Final Feedback: Estimate of time is **accurate**, in practice, I spent around 2 hours and 20 minutes finishing final comparison and observing final results (learning curves & error).

- e) **Conclusion:** Based on **Final Feedbacks** mentioned above, generally speaking, I think that the estimate of my work plan is pretty accurate; also, yes, it is a pretty good estimate because the time distribution from this estimate makes sense to me which parts I will spend more times, and which part I will not spend a lot of times, helping me more dynamically distribute my time when finishing this project (i.e., instead of spending complete part of time, e.g., 5 hours, I can divide it into small units, such as 1+2+2 hours, fully taking advantage of my spare time).

8. Testing:

- a) **part1: Test the correlation between our chosen attributes and price.**
(numerical data is stored in 'result.txt', and code for this part test is in 'data_process.py', pearson_check method & regression_line_check method)

(1) We test the correlation between our chosen attributes and price mainly through two ways: **pearson correlation coefficient** and **trend lines**.

- (2) **pearson correlation coefficient:** We compute the pearson coefficient between each chosen attribute and price data, storing the calculated result in 'result.txt'. Here is the coefficient result:

```
[('regDate', 0.598549770784627),  
 ('model', 0.13785078771346965),  
 ('bodyType', 0.06563072676118983),  
 ('creatDate', 0.24137301242451323),  
 ('v_0', 0.33855443481030517),  
 ('v_2', 0.33330804724195473),  
 ('v_3', -0.7109939870722409),  
 ('v_5', 0.09376225451302536),  
 ('v_11', -0.33592841849331256),  
 ('v_12', 0.74437072715982)]
```

For each tuple, such as ('regDate', 0.598549770784627), it means that the correlation (coefficient) between regDate attribute and price is 0.598549770784627, and the positive / negative symbol simple means positive relationship / negative relationship. Admittedly, among all the coefficient values, based on our originally setting standard "if this coefficient ≥ 0.8 , then we can say the correlation exists", this result is not very ideal. Through further reading & analysis, I think these are possible reasons for this phenomena.

- The condition coefficient ≥ 0.8 is too strong. If we observe deeper in these attributes, we can find that the maximum of absolute values of coefficients is around 0.74, and this is the strongest relationship we can find among all attributes. Therefore, based on our current situation (relationship is not so strong as we expected), we shall loosen/weaken our condition a bit (e.g., lower the required coefficient threshold), making it more practical when applying to our project.
- The pearson coefficient is more suitable for continuous data, but our given dataset is very huge, including lots of different / special cases. Therefore, for our dataset, there may exist more appropriate standards / coefficient to evaluate how strong the relationship is between chosen attributes and price.
- Observing deeper through pearson coefficient, we can find that although to some chosen attributes, their pearson coefficient is not very high, yet if we compare them with other attributes that are not chosen, we can see that overall, pearson coefficient of these chosen attributes are still higher than those unchosen attributes; therefore, to our goal "choose top 10 most important

influence factors”, what we get is still the best result, i.e., our analysis is correct.

- (3) **trend lines:** We plot the trend lines (regression lines) between each chosen attribute (x-axis) & price (y-axis). Because the number of plots is huge (10 plots), please check my submitted plots. These are plots of trend lines between each chosen attribute and price:

```
['bodyType_and_price_trend.png',  
'creatDate_and_price_trend.png',  
'model_and_price_trend.png',  
'regDate_and_price_trend.png',  
'v_0_and_price_trend.png',  
'v_2_and_price_trend.png',  
'v_3_and_price_trend.png',  
'v_5_and_price_trend.png',  
'v_11_and_price_trend.png',  
'v_12_and_price_trend.png',  
]
```

In short summary, we can see that in these plots, the general trend of regression line can show that data of each chosen attribute is **changing consistently** with data of price (**increase value of a chosen attribute with increase value of price** or **increase value of a chosen attribute with decrease value of price**). These can show that our analysis is correct.

- b) **part2: Test whether model trained by random forest is better than other traditional ML methods.** (numerical data is stored in '**validation_result.txt**', and code for this part test is in '**more_test.py**')
(1) We test which model is better through predication accuracy and error value on a smaller dataset (10000 samples); the model with higher accuracy and lower error is the best.

- (2) **linear:**

acc: 0.9666223996693466
error: 170.17331196179848

ridge:

acc: 0.6440266983508678
error: 1814.9044602219712

random forest:

acc: 0.978322032864788
error: 110.52356752591727

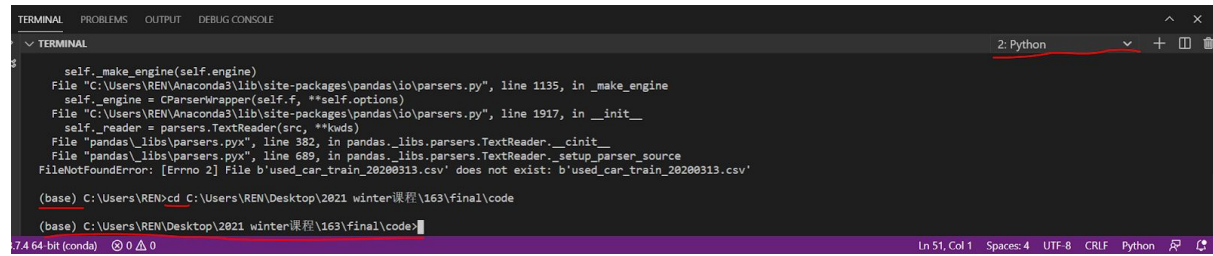
- (3) From accuracy & error values listed above, we can see that model trained by random forest method is with the highest accuracy and lowest error value, so this proves that my conclusion “the model trained by random forest is better than other traditional ML methods” is correct.

9. **Collaboration:**

- The sklearn and tensorflow documentations are very helpful in code implementation when I feel confused about specific steps in building a model with a particular method.
- StackOverflow is very helpful when I face conceptual problems about different Machine Learning methods & implementation problems when building specific models.
- Wikipedia is very helpful when I face conceptual problems regarding math knowledge to use in the project.

10. **Further Note About Code Operation & Output Files:** (combine with README.md)

- a) VScode terminal path setting: When operate the code, please use the 'Python' terminal, and set the path to the direct folder where you put / store these code files. See the Red Highlighted part below.



```
self._make_engine(self.engine)
File "C:\Users\REN\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1135, in _make_engine
self._engine = CParserWrapper(self.f, **self.options)
File "C:\Users\REN\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1917, in __init__
self._reader = parsers.TextReader(src, **kwds)
File "pandas\libs\parsers.pyx", line 382, in pandas.libs.parsers.TextReader.__cinit__
File "pandas\libs\parsers.pyx", line 689, in pandas.libs.parsers.TextReader._setup_parser_source
FileNotFoundError: [Errno 2] File b'used_car_train_20200313.csv' does not exist: b'used_car_train_20200313.csv'

(base) C:\Users\REN>cd C:\Users\REN\Desktop\2021 winter课程\163\final\code
(base) C:\Users\REN\Desktop\2021 winter课程\163\final\code>
```

- b) Among all the output files,

- ['bodyType_and_price_trend.png',
'creatDate_and_price_trend.png',
'model_and_price_trend.png',
'regDate_and_price_trend.png',
'v_0_and_price_trend.png',
'v_2_and_price_trend.png',
'v_3_and_price_trend.png',
'v_5_and_price_trend.png',
'v_11_and_price_trend.png',
'v_12_and_price_trend.png',
]

these png files store trend line plots for the Testing part.

- 'heatmap_for_correlation.png' **stores heatmap analysis result.**
- ['linear_regression_training_curve.png',
'ridge_regression_training_curve.png',
'random_forest_training_curve.png'
]

these png files store learning curve plots for the traditional Machine Learning methods.

- ['l1_relu_curve.png',
'l2_relu_curve.png',
'l2_sigmoid_curve.png'
]

these png files store learning curve plots for the Deep Learning methods.

- ['linear.model',
'ridge.model',
'random_forest.model'
]

these model files store models trained by traditional Machine Learning methods.

- 'result.txt' **stores data from data analysis & model training, including pearson correlation coefficient & errors from each model.**

- 'validation_result.txt' stores data for our testing part 2), including accuracy & error values of each model trained by traditional Machine Learning methods, tested on small dataset.