Used cars in different states

Shangyu Chen, Yuqi Yan, Ruiheng Xie

McKelvey Engineering School, Washington University in St. Louis

INFO 574: Foundation of Analytics

Prof Angelique Zeringue

Dec 16, 2022

Problem Statement

Our goal in this analysis is to find out what factors the price of a used car depends on. We have 3 Problems, which are: (1) The relationship between state personal per capita income and the state average price of used cars; (2) The relationship between used cars' mileage, year, title status, and price. (3) Can we change the predictor or use other models to get a more significant relationship?

Background

We found that during the COVID-19 pandemic, used car prices across the United States skyrocketed. The prices of used cars were much higher than before the pandemic. Also, the prices of used cars can vary depending on where they are located. Sometimes two cars in the same condition and model but in different states will have different prices. Therefore, we want to find out what factors are related to the price of a used car.

Data Source

The two datasets that we have are from Kaggle Datasets. The first dataset is the "USA_cars_dataset." It has 2499 observations. And it has some features, which include year, mileage, title status, and color. The second dataset is the "USA_states_dataset." It has 57 observations, including the USA, 50 states, D.C., and 5 territories with a permanent non-military population. It has some features: personal per capita income, household/family income median, and population. Here is a summary of these two datasets.

	price	log_price	mileage	log_mileage	year	title_status_clean
count	2356.000000	2356.000000	2.356000e+03	2356.000000	2356.000000	2356.00000
mean	19600.726231	9.677337	4.692227e+04	10.456179	2017.252971	0.96944
std	11640.568439	0.704109	4.681705e+04	0.781795	2.048170	0.17216
min	1025.000000	6.932448	1.091000e+03	6.994850	2010.000000	0.00000
25%	11100.000000	9.314700	2.128100e+04	9.965570	2016.000000	1.00000
50%	17500.000000	9.769956	3.468550e+04	10.454077	2018.000000	1.00000
75%	26000.000000	10.165852	5.701725e+04	10.951109	2019.000000	1.00000
max	84900.000000	11.349229	1.017936e+06	13.833288	2020.000000	1.00000

Figure 1: Describe of "USA_cars_datasets"

	Per capita	Personal per capita income (2020), BEA[10]	Of which disposable personal per capita income (2020), BEA[11]	Median	Median.1	Population (April 1, 2020)
count	51.000000	51.000000	51.000000	51.000000	51.000000	5.100000e+01
mean	35149.098039	57740.803922	51698.117647	65511.313725	81768.882353	6.498992e+06
std	6075.379011	9405.932513	7421.921828	11171.343911	14051.282953	7.408017e+06
min	25301.000000	42129.000000	39083.000000	45792.000000	58503.000000	5.768510e+05
25%	31653.000000	51371.500000	46684.500000	57506.000000	72689.500000	1.816411e+06
50%	33272.000000	55675.000000	49804.000000	63229.000000	79006.000000	4.505836e+06
75%	37626.500000	61981.500000	55912.000000	75028.000000	90426.000000	7.428392e+06
max	59808.000000	86567.000000	73568.000000	92266.000000	130291.000000	3.953822e+07

Figure 2: Describe of "USA_states_datasets"

Methods

- **General**

The first step is to process the raw data. First, our dataset has some non-integer-type data inside. For example, there is some data with dollar signs and commas in front of the numbers. We need to remove "$" (the dollar sign) and "," (the comma) first.

State or territory		rsonal per capita income (2020), BEA|
Alabama	$28,650	$46,479
Alaska	$36,978	$63,502
Arizona	$32,173	$49,648
Arkansas	$27,274	$47,235

	state	Per capita	Personal per capita income (2020), BEA[10]	Of which disposable personal per capita income (2020), BEA[11]
0	alabama	28650.0	46479.0	42392.0
1	alaska	36978.0	63502.0	59053.0
2	arizona	32173.0	49648.0	45025.0

Figure 3: Removal of dollar signs and commas

Then, we had unexpected data in our dataset, such as some vehicles that cost $0 and others that ran 0 miles. These are data that should not exist, and we should remove them.

	price	brand	model	year	title_status	mileage	color
309	0	chevrolet	door	2004	salvage insu	0	maroon
322	0	ford	chassis	1994	salvage insu	0	green
545	0	gmc	door	1993	salvage insu	0	light blue
504	100	peterbilt	truck	2012	salvage insu	0	blue
1619	650	ford	door	2017	salvage insu	0	black
1236	4200	ford	door	2013	clean vehicl	0	no_color
349	0	heartland	sundance	2009	clean vehicl	1	white
2417	375	nissan	door	2017	salvage insu	1	red

Figure 4: Unexpected data removal

Then we need to remove some issues in the two datasets that are not written the same way, such as the case of the first letter of each state. We need to standardize the case issue.

5	Maryland	$43,325	$66,799
6	New York	$41,857	$74,472

5	maryland	43325.0	66799.0
6	new york	41857.0	74472.0

Figure 5: Dealing with the case issue

Our second step is to process some outlier data. While we were normalizing the data, we found some outliers. For example, some vehicles are priced at just $25, and some vehicles' mileage value is only "1". Also, there are very little data on cars before 2010, but with lots of outliers – we will clear all of them. All the data should be cleaned up as well.

Thus, we removed all data for vehicles priced below $1,000 and all data for vehicles with mileage below 1,000 miles.

- **The method in Problem 1**

The first problem is to discuss the relationship between state personal per capita income and the state average price of used cars. Get information on average car prices per state, etc.

```
df_state.rename(columns={'State or territory':'state'}, inplace=True)
df_state_gdp = df_state.groupby(by=['state']).mean().reset_index()
df_state_gdp.head(5)
```

	state	Per capita	Personal per capita income (2020), BEA[10]	Of which disposable personal per capita income (2020), BEA[11]	Median	Median.1	Population (April 1, 2020)
0	alabama	28650.0	46479.0	42392.0	51734.0	66171.0	5024279.0
1	alaska	36978.0	63502.0	59053.0	75463.0	91971.0	733391.0
2	arizona	32173.0	49648.0	45025.0	62055.0	74468.0	7151502.0
3	arkansas	27274.0	47235.0	43083.0	48952.0	62387.0	3011524.0
4	california	39393.0	70192.0	60796.0	80440.0	91377.0	39538223.0

Figure 6: State information

```
df_car_state = df_car_datasets.groupby(by=['state']).mean().reset_index()
df_car_state.head(5)
```

	state	price	year	mileage	lot	log_price	log_mileage	year2	title_status_clean	year2_	log_mileage_	title_status_cl
0	alabama	23872.058824	2016.352941	64916.176471	1.677416e+08	9.779967	10.912924	43.882353	1.000000	-0.506724	0.584226	0.17
1	arizona	15926.666667	2017.833333	31805.100000	1.675652e+08	9.526146	10.052841	65.500000	1.000000	0.341356	-0.515912	0.17
2	arkansas	7435.000000	2014.166667	72893.166667	1.676102e+08	8.680840	11.109932	23.500000	0.500000	-1.306343	0.836221	-2.72
3	california	18583.333333	2017.616667	40240.483333	1.676697e+08	9.619421	10.420709	60.616667	0.972222	0.149778	-0.045370	0.01
4	colorado	18992.857143	2016.500000	50236.000000	1.676291e+08	9.776871	10.526250	46.642857	1.000000	-0.398427	0.089629	0.17

Figure 7: Average price information

Some states have no information about used car prices in the data frame, showing "NaN" in the data. We should delete these states.

	state	price	Personal per capita income (2020), BEA[10]
0	alabama	23872.058824	46479.0
1	alaska	NaN	63502.0
2	arizona	15926.666667	49648.0
3	arkansas	7435.000000	47235.0
4	california	18583.333333	70192.0

Figure 8: Data without information

Combine the two tables and standardize the price and personal income per capita information using z-score.

```
df_all = pd.merge(df_car_state, df_state_gdp)
df_all["s_price"] = (df_all["price"] - np.mean(df_all["price"])) / np.std(df_all["price"])
df_all["s_Personal per capita income"] = (df_all["Personal per capita income (2020), BEA[10]"] -
                                          np.mean(df_all["Personal per capita income (2020), BEA[10]"])
                                         ) / np.std(df_all["Personal per capita income (2020), BEA[10]"])
df_all.head(5)
```

	state	price	year	mileage	lot	log_price	log_mileage	year2	title_status_clean	year2_	log_mileage_	title_status_cl
0	alabama	23872.058824	2016.352941	64916.176471	1.677416e+08	9.779967	10.912924	43.882353	1.000000	-0.506724	0.584226	0.17
1	arizona	15926.666667	2017.833333	31805.100000	1.675652e+08	9.526146	10.052841	65.500000	1.000000	0.341356	-0.515912	0.17
2	arkansas	7435.000000	2014.166667	72893.166667	1.676102e+08	8.680840	11.109932	23.500000	0.500000	-1.306343	0.836221	-2.72
3	california	18583.333333	2017.616667	40240.483333	1.676697e+08	9.619421	10.420709	60.616667	0.972222	0.149778	-0.045370	0.01
4	colorado	18992.857143	2016.500000	50236.000000	1.676291e+08	9.776871	10.526250	46.642857	1.000000	-0.398427	0.089629	0.17

Figure 9: Information combination

View state, price, and per capita personal income information in the combined data frame (The empty value rows have been removed), resulting in 42 observations.

```
df_all.loc[:,["state", "s_price","s_Personal per capita income"]].head(5)
```

	state	s_price	s_Personal per capita income
0	alabama	0.648215	-1.147368
1	arizona	-0.344606	-0.798301
2	arkansas	-1.405688	-1.064094
3	california	-0.012641	1.464633
4	colorado	0.038531	0.757907

Figure 10: Filter data

View the normalized price distribution; it basically fits the normal distribution and does not need to be transformed.

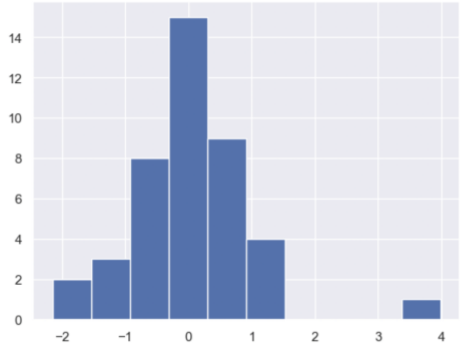

Figure 11: Normalized price distribution

Scatter plots were created using prices and per capita personal income for 42 observations. It seems that there is no linear relationship.

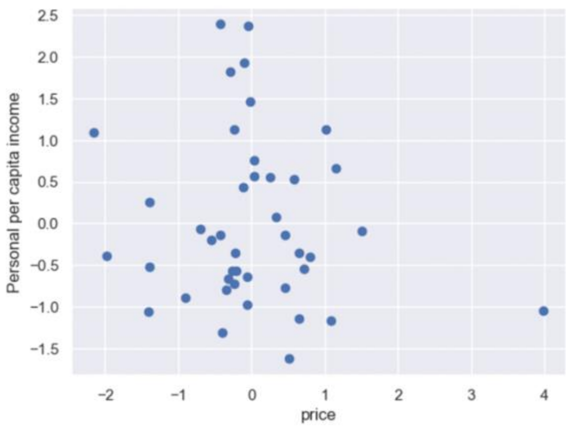

Figure 12: Scatter plots

- **The method in Problem 2:**

The second question discusses the relationship between used cars' mileage, year, title status, color, and price.

First, we looked at the histograms for the two categories of price and mileage. We found that these two categories are generally not distributed.

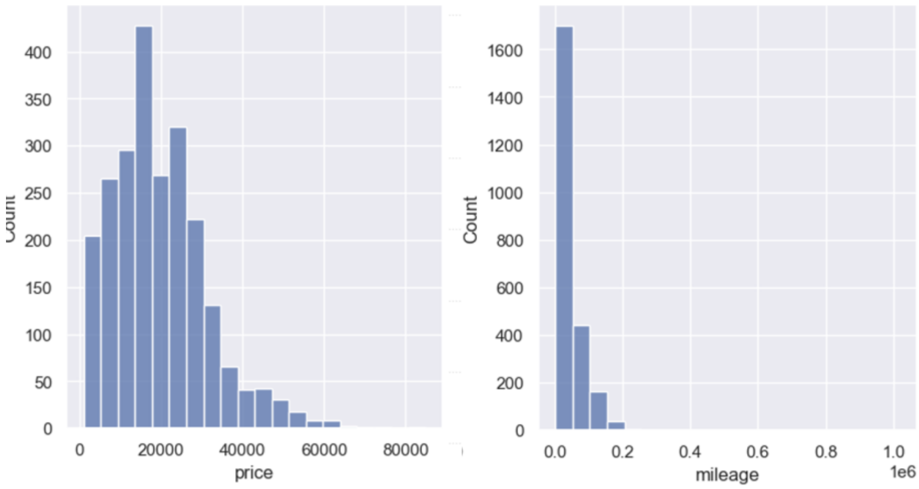

Figure 13: Histogram plots of price and mileage

We use np.log to make the data normally distributed.

```
# Make the data looks normal
df_car_datasets.loc[:, 'log_price'] = np.log(df_car_datasets.loc[:, 'price'])
df_car_datasets.loc[:, 'log_mileage'] = np.log(df_car_datasets.loc[:, 'mileage'])
df_car_datasets.loc[:, 'year2'] = (df_car_datasets.loc[:, 'year'] - 2010) ** 2
```

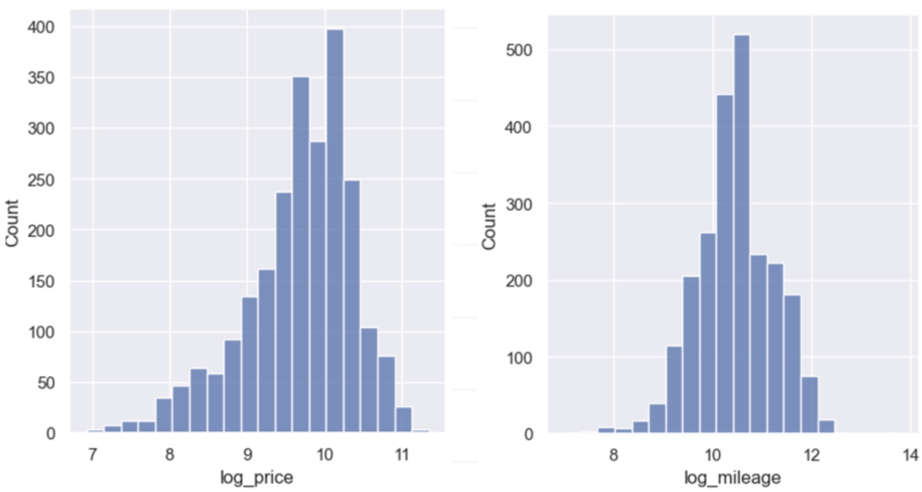

Figure 14: The logged data

Then we use Multiple Linear Regression to find the relation between price and these categories.

OLS Regression Results

Dep. Variable:	log_price	**R-squared:**	0.365
Model:	OLS	**Adj. R-squared:**	0.363
Method:	Least Squares	**F-statistic:**	168.6
Date:	Thu, 15 Dec 2022	**Prob (F-statistic):**	5.60e-225
Time:	15:42:02	**Log-Likelihood:**	-1981.2
No. Observations:	2356	**AIC:**	3980.
Df Residuals:	2347	**BIC:**	4032.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-126.1008	15.821	-7.970	0.000	-157.126	-95.075
log_mileage	-0.2922	0.020	-14.521	0.000	-0.332	-0.253
year	0.0684	0.008	8.785	0.000	0.053	0.084
title_status_clean	1.0337	0.070	14.760	0.000	0.896	1.171
color_white	0.0185	0.030	0.608	0.543	-0.041	0.078
color_blue	-0.1558	0.052	-2.994	0.003	-0.258	-0.054
color_red	-0.0711	0.047	-1.498	0.134	-0.164	0.022
color_gray	-0.1763	0.036	-4.887	0.000	-0.247	-0.106
color_silver	-0.1549	0.040	-3.907	0.000	-0.233	-0.077

Omnibus:	60.074	**Durbin-Watson:**	1.728
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	67.445
Skew:	-0.356	**Prob(JB):**	2.26e-15
Kurtosis:	3.424	**Cond. No.**	2.76e+06

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.76e+06. This might indicate that there are
strong multicollinearity or other numerical problems.

Figure 15: OLS Regression Results

We found the R-square = 0.365, i.e., only 36.5% of the data were clarified. Then, we got the VIF value to check the multilinear relationship.

	variable	VIF
0	const	1.866704e+06
1	log_mileage	1.844541e+00
2	year	1.893001e+00
3	title_status_clean	1.083712e+00
4	color_white	1.409224e+00
5	color_blue	1.128311e+00
6	color_red	1.161095e+00
7	color_gray	1.296634e+00
8	color_silver	1.252921e+00

Figure 16: VIF

VIF values of the mileage, year, “status_clean”, and color (white, blue, red, gray, and silver) are close to 1. There is almost no multicollinearity.

- **The method in Problem 3:**

In the result of the model summary of problem 2, we found that the condition number is too large -- we cannot easily say that we can predict the price simply from some features.

We first transformed the feature "year," making the year data more like normal distributions.

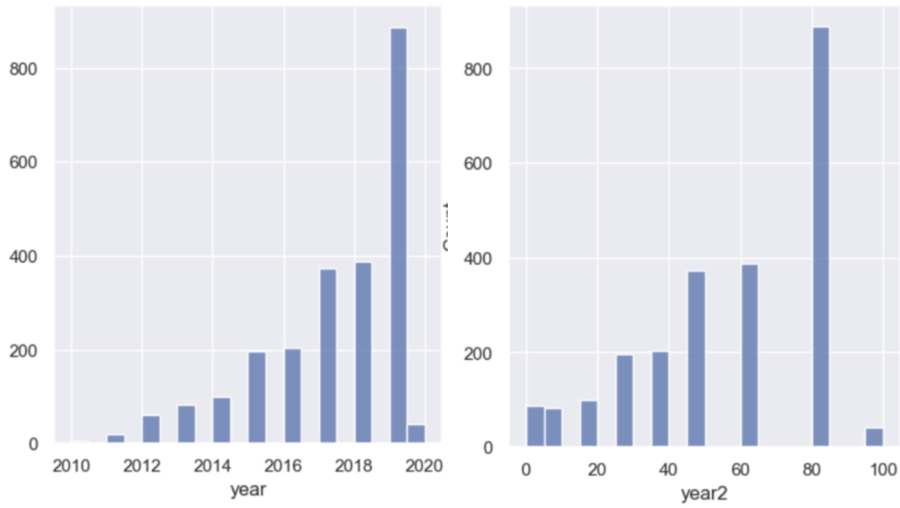

Figure 17: Year data transformation

Another obvious problem of the problem 2 model is that the price of a car can be affected by the brand of the car - for example, SUVs with the same mileage and age tend to be more expensive than smaller vehicles. Since there were vehicles with less data, we decided to filter only two brands of cars for analysis. We found that the brand Ford and Chevrolet tend to have more data; we can choose only 2 specific brands to analyze the data. As a result, we will add "brand" as a predictor and get 1442 observations.

We also need to know if a variable is helpful for prediction. We solve this problem by removing or adding variables as predictors.

We first use log mileage, transformed year data, title status, and brand code as predictors. We found that the R^2 value is 0.381, and the condition number is small.

OLS Regression Results

Dep. Variable:	log_price	**R-squared:**	0.379
Model:	OLS	**Adj. R-squared:**	0.377
Method:	Least Squares	**F-statistic:**	175.0
Date:	Thu, 15 Dec 2022	**Prob (F-statistic):**	4.89e-117
Time:	15:42:02	**Log-Likelihood:**	-935.72
No. Observations:	1152	**AIC:**	1881.
Df Residuals:	1147	**BIC:**	1907.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	9.8160	0.039	254.422	0.000	9.740	9.892
log_mileage_	-0.1615	0.022	-7.492	0.000	-0.204	-0.119
year2_	0.2033	0.023	9.036	0.000	0.159	0.247
title_status_clean_	0.1678	0.016	10.348	0.000	0.136	0.200
brand_ford	0.0048	0.042	0.112	0.911	-0.079	0.088

Omnibus:	34.227	**Durbin-Watson:**	2.136
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	36.578
Skew:	-0.419	**Prob(JB):**	1.14e-08
Kurtosis:	3.244	**Cond. No.**	4.92

Figure 18: Only 4 predictors

We then add the average income (the variable GDP is the average income) and color data as predictors, finding that R-square may stay the same, especially when adding colors.

(figure in the next page)

OLS Regression Results

Dep. Variable:	log_price	**R-squared:**	0.386
Model:	OLS	**Adj. R-squared:**	0.383
Method:	Least Squares	**F-statistic:**	144.1
Date:	Thu, 15 Dec 2022	**Prob (F-statistic):**	9.96e-119
Time:	15:42:02	**Log-Likelihood:**	-929.12
No. Observations:	1152	**AIC:**	1870.
Df Residuals:	1146	**BIC:**	1901.
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	9.8359	0.039	253.739	0.000	9.760	9.912
log_mileage_	-0.1505	0.022	-6.949	0.000	-0.193	-0.108
year2_	0.2173	0.023	9.568	0.000	0.173	0.262
title_status_clean_	0.1656	0.016	10.257	0.000	0.134	0.197
brand_ford	-0.0181	0.043	-0.424	0.671	-0.102	0.066
gdp_	0.0602	0.017	3.635	0.000	0.028	0.093

Omnibus:	26.231	**Durbin-Watson:**	2.119
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	27.501
Skew:	-0.360	**Prob(JB):**	1.07e-06
Kurtosis:	3.231	**Cond. No.**	4.98

OLS Regression Results

Dep. Variable:	log_price	**R-squared:**	0.390
Model:	OLS	**Adj. R-squared:**	0.385
Method:	Least Squares	**F-statistic:**	73.03
Date:	Thu, 15 Dec 2022	**Prob (F-statistic):**	2.76e-115
Time:	15:42:02	**Log-Likelihood:**	-925.14
No. Observations:	1152	**AIC:**	1872.
Df Residuals:	1141	**BIC:**	1928.
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	9.8535	0.045	217.946	0.000	9.765	9.942
log_mileage_	-0.1493	0.022	-6.891	0.000	-0.192	-0.107
year2_	0.2183	0.023	9.606	0.000	0.174	0.263
title_status_clean_	0.1654	0.016	10.211	0.000	0.134	0.197
brand_ford	-0.0223	0.043	-0.521	0.602	-0.106	0.062
gdp_	0.0612	0.017	3.679	0.000	0.029	0.094
color_white	-0.0333	0.041	-0.811	0.417	-0.114	0.047
color_blue	-0.0964	0.084	-1.150	0.250	-0.261	0.068
color_red	0.0816	0.062	1.319	0.187	-0.040	0.203
color_gray	-0.0771	0.050	-1.542	0.123	-0.175	0.021
color_silver	0.0320	0.056	0.574	0.566	-0.077	0.141

Figure 19: Add some features, showing that is not significant

We can see that color is not a significant predictor. So we use predictors mileage, year, title status, brand, and average income to make predictions.

Then we will test different models: Linear regression, Ridge (regularized linear regression), single decision tree, and random forest. We don’t use Lasso because this will erase some critical features.

Here are R^2, test scores, and mean squared errors of different models:

Model	R^2	Test scores	Mean squared error
Linear regression	0.388	0.352	0.321
Ridge	0.388	0.365	0.309
Single decision tree	1.0	-0.108	0.540
Random forest	0.693	0.441	0.277

The special thing is: when we use the random forest model, we use cross validation to calculate the training score to find out the better decision tree’s maximum depth. We got the cross validation plot as follows:

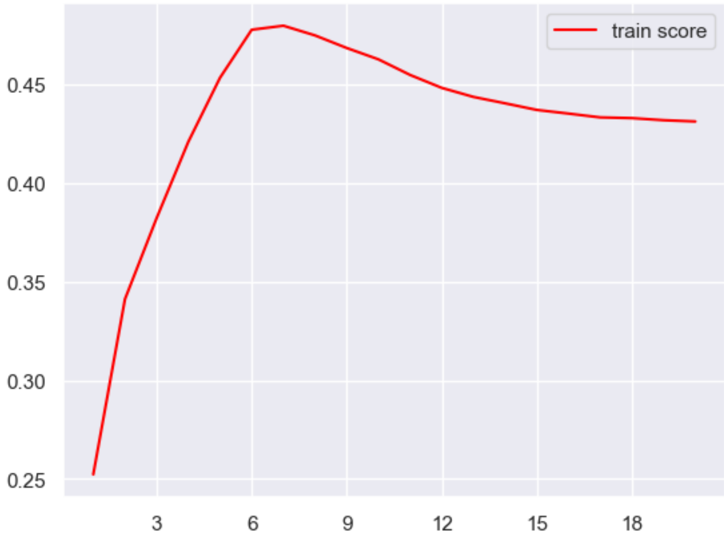

Figure 20: Cross-validation score plot

It is meaningless to calculate R^2 of a single decision tree and random forest because the decision tree can perfectly fit the data. However, the random forest model shows better generalized performance, for it has better test scores and fewer means squared errors.

Using mileage, year, title status, brand, and average income as predictors is better. Under this condition, using the random forest model is better.

Result

There is no relationship between state personal per capita income and the state average price of cars.

Also, if we don't consider car brands, we found little relationship between used cars' mileage year, title status, and price.

Using mileage, year, title status, brand, and the state's average income as predictors is better.

By the way, the "random forest" model here is better than the linear model but still not stable.

Discussion

Although the random forest has a good R^2 on the train set, we still cannot say that the random forest is better than the R^2 because of the property of the decision tree. The decision tree can perfectly fit the data even if we make some limitations on its max depth. However, in this case, when we calculate the mean squared error in the decision tree, we found that decision tree bagging still has better generalized performance than the linear model. That is because ensembled learners are better than the single learner.

However, the model is still not stable. First, as we know, even in a specific model, cars have different setups. Take the Toyota RAV4 2023 as an example, Toyota RAV4 LE has less price than XLE. However, we are caught in a dilemma here. We chose specific 2 car brands instead of specific car models for 2 reasons: if we choose specific car models, we will create more features and be likely to get fewer data. As a result, the model will be overfitting.

In the process of data analysis, it is often difficult to get the desired solution in one step. We should get more data apart from optimizing our models. In this analysis of "predicting used car prices," we did not get a good prediction model because some key data affecting used car prices were still missing.

References

- Dataset: https://www.kaggle.com/datasets
- OLS regression: https://www.statsmodels.org/dev/generated/statsmodels.regression.linear_model.OLS.html
- VIF: https://www.statology.org/how-to-calculate-vif-in-python/
- Random Forest: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html