T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

CAR PRICE PREDICTION USING REGRESSION

SENA ERDOĞAN

SUPERVISOR
ASSOC. PROF. DR. BURCU YILMAZ

GEBZE
2023

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CAR PRICE PREDICTION USING REGRESSION

SENA ERDOĞAN

SUPERVISOR
ASSOC. PROF. DR. BURCU YILMAZ

2023
GEBZE

ABSTRACT

This project is about predicting what a car's price is going to be in the upcoming years according to its many features using regression.

Keywords: regression, linear regression, logistic regression, random forest regression

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The dataset I'm using consists of 50000 rows and 13 columns. I used many different parts of this dataset to get a higher accuracy using the same model. I ended up using 5000, 25000 and 50000 rows of the dataset, separately and compare the accuracies.

The dataset's features are price, model year, model, condition, cylinders, fuel, odometer, transmission, type, paint color, is 4wd, date posted and days listed.

I used regression to solve this problem since it would be the most suitable technique for the solution.

Regression is a supervised machine learning technique that is used for predicting a continuous target variable based on one or more input features. It is used to model the relationship between the input features and the target variable, and it can be used to understand and make predictions about the underlying processes that generate the data.

The demo link: My Demo.

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	2019-02-07	79
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	2019-04-02	28
5	14990	2014.0	chrysler 300	excellent	6.0	gas	57954.0	automatic	sedan	black	2018-06-20	15
6	12990	2015.0	toyota camry	excellent	4.0	gas	79212.0	automatic	sedan	white	2018-12-27	73
7	15990	2013.0	honda pilot	excellent	6.0	gas	109473.0	automatic	SUV	black	2019-01-07	68
...	...	...	...	...	...	...	...	...	...	...	...	...
5091	7300	2013.0	nissan altima	excellent	6.0	gas	7100.0	automatic	sedan	black	2018-06-17	63
5092	17999	2014.0	ram 1500	like new	8.0	gas	154000.0	automatic	pickup	white	2018-06-21	11
5093	4800	2012.0	volkswagen jetta	good	4.0	gas	138000.0	automatic	sedan	silver	2018-07-19	39
5094	12000	2005.0	chevrolet silverado 2500hd	good	8.0	diesel	228000.0	automatic	pickup	silver	2018-08-18	52
5096	11000	2012.0	dodge charger	excellent	8.0	gas	81000.0	automatic	sedan	black	2019-04-19	44

3031 rows × 12 columns

Figure 1.1: logistic regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	61030.0	automatic	truck	black	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	17037.0	automatic	bus	white	2018-08-20	73
5	7000	2002.0	toyota tacoma	fair	6.0	gas	128200.0	automatic	truck	silver	2018-08-09	24
6	3900	2003.0	jeep liberty	good	4.0	gas	128872.0	automatic	SUV	brown	2018-06-26	40
10	8000	2008.0	ford f-250 super duty	good	10.0	gas	240000.0	manual	truck	red	2019-04-13	49
...	...	...	...	...	...	...	...	...	...	...	...	...
25136	17980	2014.0	jeep grand cherokee	good	6.0	gas	100255.0	automatic	SUV	white	2019-02-27	77
25137	15980	2015.0	ford explorer	good	8.0	gas	122654.0	automatic	SUV	black	2018-05-23	25
25138	29999	2011.0	ford econoline	like new	10.0	gas	117000.0	automatic	van	white	2019-04-09	59
25140	25200	2014.0	chevrolet silverado 1500 crew	good	6.0	gas	85048.0	automatic	pickup	silver	2019-04-06	54
25141	2650	2006.0	toyota sienna	good	6.0	gas	202000.0	automatic	mini-van	red	2018-11-24	47

14550 rows × 12 columns

Figure 1.2: logistic regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	2019-02-07	79
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	2019-04-02	28
5	14990	2014.0	chrysler 300	excellent	6.0	gas	57954.0	automatic	sedan	black	2018-06-20	15
6	12990	2015.0	toyota camry	excellent	4.0	gas	79212.0	automatic	sedan	white	2018-12-27	73
7	15990	2013.0	honda pilot	excellent	6.0	gas	109473.0	automatic	SUV	black	2019-01-07	68
...	...	...	...	...	...	...	...	...	...	...	...	...
51518	3750	2005.0	ford taurus	excellent	6.0	gas	110200.0	automatic	sedan	silver	2018-08-10	63
51520	9249	2013.0	nissan maxima	like new	6.0	gas	88136.0	automatic	sedan	black	2018-10-03	37
51521	2700	2002.0	honda civic	salvage	4.0	gas	181500.0	automatic	sedan	white	2018-11-14	22
51522	3950	2009.0	hyundai sonata	excellent	4.0	gas	128000.0	automatic	sedan	blue	2018-11-15	32
51523	7455	2013.0	toyota corolla	good	4.0	gas	139573.0	automatic	sedan	black	2018-07-02	71

29916 rows × 12 columns

Figure 1.3: logistic regression 50000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 1.4: linear regression 5000 head

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
5092	17999	2014.0	ram 1500	like new	8.0	gas	154000.0	automatic	pickup	white	1.0	2018-06-21	11
5093	4800	2012.0	volkswagen jetta	good	4.0	gas	138000.0	automatic	sedan	silver	NaN	2018-07-19	39
5094	12000	2005.0	chevrolet silverado 2500hd	good	8.0	diesel	228000.0	automatic	pickup	silver	1.0	2018-08-18	52
5095	8999	2011.0	jeep grand cherokee	good	6.0	gas	NaN	automatic	SUV	white	1.0	2018-11-01	51
5096	11000	2012.0	dodge charger	excellent	8.0	gas	81000.0	automatic	sedan	black	NaN	2019-04-19	44

Figure 1.5: linear regression 5000 tail

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	59995	NaN	ram 2500	like new	6.0	diesel	22635.0	automatic	truck	white	1.0	2018-06-02	52
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	61030.0	automatic	truck	black	1.0	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	17037.0	automatic	bus	white	NaN	2018-08-20	73
3	2850	1997.0	ford f150	good	8.0	gas	255040.0	manual	pickup	NaN	1.0	2018-06-05	18
4	3700	2007.0	honda accord	fair	NaN	gas	173600.0	manual	sedan	red	NaN	2018-10-02	26

Figure 1.6: linear regression 25000 head

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
25139	12000	2004.0	jeep wrangler	excellent	6.0	gas	130000.0	automatic	offroad	NaN	1.0	2018-10-15	8
25140	25200	2014.0	chevrolet silverado 1500 crew	good	6.0	gas	85048.0	automatic	pickup	silver	1.0	2019-04-06	54
25141	2650	2006.0	toyota sienna	good	6.0	gas	202000.0	automatic	mini-van	red	NaN	2018-11-24	47
25142	12999	2012.0	chevrolet colorado	good	4.0	gas	92816.0	automatic	truck	NaN	1.0	2018-09-12	61
25143	1000	1997.0	ford expedition	fair	8.0	gas	NaN	automatic	SUV	NaN	1.0	2018-06-01	113

Figure 1.7: linear regression 25000 tail

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 1.8: linear regression 50000 head

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
51520	9249	2013.0	nissan maxima	like new	6.0	gas	88136.0	automatic	sedan	black	NaN	2018-10-03	37
51521	2700	2002.0	honda civic	salvage	4.0	gas	181500.0	automatic	sedan	white	NaN	2018-11-14	22
51522	3950	2009.0	hyundai sonata	excellent	4.0	gas	128000.0	automatic	sedan	blue	NaN	2018-11-15	32
51523	7455	2013.0	toyota corolla	good	4.0	gas	139573.0	automatic	sedan	black	NaN	2018-07-02	71
51524	6300	2014.0	nissan altima	good	4.0	gas	NaN	automatic	sedan	NaN	NaN	2018-06-05	10

Figure 1.9: linear regression 50000 tail

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 1.10: random forest regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	59995	NaN	ram 2500	like new	6.0	diesel	22635.0	automatic	truck	white	1.0	2018-06-02	52
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	61030.0	automatic	truck	black	1.0	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	17037.0	automatic	bus	white	NaN	2018-08-20	73
3	2850	1997.0	ford f150	good	8.0	gas	255040.0	manual	pickup	NaN	1.0	2018-06-05	18
4	3700	2007.0	honda accord	fair	NaN	gas	173600.0	manual	sedan	red	NaN	2018-10-02	26

Figure 1.11: random forest regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 1.12: random forest regression 50000

```
(51525, 13)
```

Figure 1.13: random forest regression 50000 size

2. LOGISTIC REGRESSION

Logistic Regression is a supervised machine learning algorithm that is used for solving binary classification problems. It is a generalized linear model that is used to model the probability of a certain class or event occurring given the values of the input features.

The basic idea behind logistic regression is to find the best fitting model that describes the relationship between the input features and the target variable (binary outcome). The model is represented by the logistic function, also known as the sigmoid function, which maps the input features to a probability value between 0 and 1.

This is the first algorithm that I tried to solve the problem with.

The accuracy started to decrease as the dataset started to grow. The accuracy was already low for this technique and it was clear that it was not the best method to solve this problem with. I tried to use a bigger dataset to increase the accuracy but it didn't help much.

The dataset contained missing values (NaN values) at the beginning and it's best to get rid of the missing values before doing regression on a dataset since it will mess with the accuracy. I considered two ways to get rid of the missing values:

First method is filling in the missing values. This method involves replacing the missing values with a specific value such as the mean, median, or mode of the column, or a custom value. This can be done using the fillna() function in pandas. However, it's important to make sure that the chosen value is representative of the data and that it doesn't introduce bias. I tried to use this method by replacing the missing values with the median of the related feature but I ran into a problem with python and couldn't get the median of objects or the string values. So I tried to use the second method.

The second method is dropping the rows or columns. This method involves removing the rows or columns that contain missing values. This can be done using the dropna() function in pandas. However, this method can lead to a loss of information and may not be suitable if the missing values are a significant portion of the dataset. In my case, this method did not get rid of the majority of the dataset and it actually increased accuracy for me so I went with this second method as the first step of my

preprocessing of this dataset.

The followings are the datasets I was left with after dropping the rows or columns.

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	2019-02-07	79
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	2019-04-02	28
5	14990	2014.0	chrysler 300	excellent	6.0	gas	57954.0	automatic	sedan	black	2018-06-20	15
6	12990	2015.0	toyota camry	excellent	4.0	gas	79212.0	automatic	sedan	white	2018-12-27	73
7	15990	2013.0	honda pilot	excellent	6.0	gas	109473.0	automatic	SUV	black	2019-01-07	68
...	...	...	...	...	...	...	...	...	...	...	...	...
5091	7300	2013.0	nissan altima	excellent	6.0	gas	7100.0	automatic	sedan	black	2018-06-17	63
5092	17999	2014.0	ram 1500	like new	8.0	gas	154000.0	automatic	pickup	white	2018-06-21	11
5093	4800	2012.0	volkswagen jetta	good	4.0	gas	138000.0	automatic	sedan	silver	2018-07-19	39
5094	12000	2005.0	chevrolet silverado 2500hd	good	8.0	diesel	228000.0	automatic	pickup	silver	2018-08-18	52
5096	11000	2012.0	dodge charger	excellent	8.0	gas	81000.0	automatic	sedan	black	2019-04-19	44

3031 rows × 12 columns

Figure 2.1: logistic regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	61030.0	automatic	truck	black	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	17037.0	automatic	bus	white	2018-08-20	73
5	7000	2002.0	toyota tacoma	fair	6.0	gas	128200.0	automatic	truck	silver	2018-08-09	24
6	3900	2003.0	jeep liberty	good	4.0	gas	128872.0	automatic	SUV	brown	2018-06-26	40
10	8000	2008.0	ford f-250 super duty	good	10.0	gas	240000.0	manual	truck	red	2019-04-13	49
...	...	...	...	...	...	...	...	...	...	...	...	...
25136	17980	2014.0	jeep grand cherokee	good	6.0	gas	100255.0	automatic	SUV	white	2019-02-27	77
25137	15980	2015.0	ford explorer	good	8.0	gas	122654.0	automatic	SUV	black	2018-05-23	25
25138	29999	2011.0	ford econoline	like new	10.0	gas	117000.0	automatic	van	white	2019-04-09	59
25140	25200	2014.0	chevrolet silverado 1500 crew	good	6.0	gas	85048.0	automatic	pickup	silver	2019-04-06	54
25141	2650	2006.0	toyota sienna	good	6.0	gas	202000.0	automatic	mini-van	red	2018-11-24	47

14550 rows × 12 columns

Figure 2.2: logistic regression 25000

A car's price is the target value here, so price column is now target column.

The followings are the number of non-null data we have after the preprocessing.

The followings are the final datasets to be used for the regression.

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	2019-02-07	79
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	2019-04-02	28
5	14990	2014.0	chrysler 300	excellent	6.0	gas	57954.0	automatic	sedan	black	2018-06-20	15
6	12990	2015.0	toyota camry	excellent	4.0	gas	79212.0	automatic	sedan	white	2018-12-27	73
7	15990	2013.0	honda pilot	excellent	6.0	gas	109473.0	automatic	SUV	black	2019-01-07	68
...	...	...	...	...	...	...	...	...	...	...	...	...
51518	3750	2005.0	ford taurus	excellent	6.0	gas	110200.0	automatic	sedan	silver	2018-08-10	63
51520	9249	2013.0	nissan maxima	like new	6.0	gas	88136.0	automatic	sedan	black	2018-10-03	37
51521	2700	2002.0	honda civic	salvage	4.0	gas	181500.0	automatic	sedan	white	2018-11-14	22
51522	3950	2009.0	hyundai sonata	excellent	4.0	gas	128000.0	automatic	sedan	blue	2018-11-15	32
51523	7455	2013.0	toyota corolla	good	4.0	gas	139573.0	automatic	sedan	black	2018-07-02	71

29916 rows × 12 columns

Figure 2.3: logistic regression 50000

```
<bound method IndexOpsMixin.value_counts of 2         5500
4        14900
5        14990
6        12990
7        15990
         ...
5091      7300
5092     17999
5093      4800
5094     12000
5096     11000
Name: target, Length: 3031, dtype: int64>
```

Figure 2.4: logistic regression 5000

```
2         28990
5          7000
6          3900
10         8000
          ...
25136     17980
25137     15980
25138     29999
25140     25200
25141      2650
Name: target, Length: 14550, dtype: int64>
```

Figure 2.5: logistic regression 25000

```
<bound method IndexOpsMixin.value_counts of 2         5500
4        14900
5        14990
6        12990
7        15990
         ...
51518     3750
51520     9249
51521     2700
51522     3950
51523     7455
Name: target, Length: 29916, dtype: int64>
```

Figure 2.6: logistic regression 50000

```
Int64Index: 3031 entries, 2 to 5096
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   target        3031 non-null   int64
 1   model_year    3031 non-null   float64
 2   model         3031 non-null   object
 3   condition     3031 non-null   object
 4   cylinders     3031 non-null   float64
 5   fuel          3031 non-null   object
 6   odometer      3031 non-null   float64
 7   transmission  3031 non-null   object
 8   type          3031 non-null   object
 9   paint_color   3031 non-null   object
 10  date_posted   3031 non-null   object
 11  days_listed   3031 non-null   int64
dtypes: float64(3), int64(2), object(7)
memory usage: 307.8+ KB
```

Figure 2.7: logistic regression 5000

```
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   target         14550 non-null  int64
 1   model_year     14550 non-null  float64
 2   model          14550 non-null  object
 3   condition      14550 non-null  object
 4   cylinders      14550 non-null  float64
 5   fuel           14550 non-null  object
 6   odometer       14550 non-null  float64
 7   transmission   14550 non-null  object
 8   type           14550 non-null  object
 9   paint_color    14550 non-null  object
 10  date_posted    14550 non-null  object
 11  days_listed    14550 non-null  int64
dtypes: float64(3), int64(2), object(7)
memory usage: 1.4+ MB
```

Figure 2.8: logistic regression 25000

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 29916 entries, 2 to 51523
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   target         29916 non-null  int64
 1   model_year     29916 non-null  float64
 2   model          29916 non-null  object
 3   condition      29916 non-null  object
 4   cylinders      29916 non-null  float64
 5   fuel           29916 non-null  object
 6   odometer       29916 non-null  float64
 7   transmission   29916 non-null  object
 8   type           29916 non-null  object
 9   paint_color    29916 non-null  object
 10  date_posted    29916 non-null  object
 11  days_listed    29916 non-null  int64
dtypes: float64(3), int64(2), object(7)
memory usage: 3.0+ MB
```

Figure 2.9: logistic regression 50000

	target	model_year	cylinders	odometer	days_listed	model_bmw x5	model_buick enclave	model_cadillac escalade	model_chevrolet camaro	model_chevrolet camaro lt coupe 2d	...	date_posted_2019-04-10
2	5500	2013.0	4.0	110000.0	79	0	0	0	0	0	...	0
4	14900	2017.0	4.0	80903.0	28	0	0	0	0	0	...	0
5	14990	2014.0	6.0	57954.0	15	0	0	0	0	0	...	0
6	12990	2015.0	4.0	79212.0	73	0	0	0	0	0	...	0
7	15990	2013.0	6.0	109473.0	68	0	0	0	0	0	...	0
...	...	...	...	...	...	...	...	...	...	...	...	...
5091	7300	2013.0	6.0	7100.0	63	0	0	0	0	0	...	0
5092	17999	2014.0	8.0	154000.0	11	0	0	0	0	0	...	0
5093	4800	2012.0	4.0	138000.0	39	0	0	0	0	0	...	0
5094	12000	2005.0	8.0	228000.0	52	0	0	0	0	0	...	0
5096	11000	2012.0	8.0	81000.0	44	0	0	0	0	0	...	0

3031 rows × 489 columns

Figure 2.10: logistic regression 5000

	target	model_year	cylinders	odometer	days_listed	model_bmw x5	model_buick enclave	model_cadillac escalade	model_chevrolet camaro	model_chevrolet camaro lt coupe 2d	...	date_posted_2019-04-10
1	15990	2001.0	8.0	61030.0	13	0	0	0	0	0	...	0
2	28990	2011.0	10.0	17037.0	73	0	0	0	0	0	...	0
5	7000	2002.0	6.0	128200.0	24	0	0	0	0	0	...	0
6	3900	2003.0	4.0	128872.0	40	0	0	0	0	0	...	0
10	8000	2008.0	10.0	240000.0	49	0	0	0	0	0	...	0
...	...	...	...	...	...	...	...	...	...	...	...	...
25136	17980	2014.0	6.0	100255.0	77	0	0	0	0	0	...	0
25137	15980	2015.0	8.0	122654.0	25	0	0	0	0	0	...	0
25138	29999	2011.0	10.0	117000.0	59	0	0	0	0	0	...	0
25140	25200	2014.0	6.0	85048.0	54	0	0	0	0	0	...	0
25141	2650	2006.0	6.0	202000.0	47	0	0	0	0	0	...	0

14550 rows × 490 columns

Figure 2.11: logistic regression 25000

For this regression, the random seed is 888 and the test size will be 20% of the dataset.

After training and testing the datasets, the accuracies are as follows:

In conclusion, logical regression did not give good results.

	target	model_year	cylinders	odometer	days_listed	model_bmw x5	model_buick enclave	model_cadillac escalade	model_chevrolet camaro	model_chevrolet camaro lt coupe 2d	...	date_posted_2019-04-10
2	5500	2013.0	4.0	110000.0	79	0	0	0	0	0	...	0
4	14900	2017.0	4.0	80903.0	28	0	0	0	0	0	...	0
5	14990	2014.0	6.0	57954.0	15	0	0	0	0	0	...	0
6	12990	2015.0	4.0	79212.0	73	0	0	0	0	0	...	0
7	15990	2013.0	6.0	109473.0	68	0	0	0	0	0	...	0
...	...	...	...	...	...	...	...	...	...	...	...	...
51518	3750	2005.0	6.0	110200.0	63	0	0	0	0	0	...	0
51520	9249	2013.0	6.0	88136.0	37	0	0	0	0	0	...	0
51521	2700	2002.0	4.0	181500.0	22	0	0	0	0	0	...	0
51522	3950	2009.0	4.0	128000.0	32	0	0	0	0	0	...	0
51523	7455	2013.0	4.0	139573.0	71	0	0	0	0	0	...	0

29916 rows × 490 columns

Figure 2.12: logistic regression 50000

```
['model_year', 'cylinders', 'odometer', 'days_listed']
['condition_fair', 'condition_good', 'condition_like new', 'condition_new', 'condition_salvage', 'date_posted_2018-05
-02', 'date_posted_2018-05-03', 'date_posted_2018-05-04', 'date_posted_2018-05-05', 'date_posted_2018-05-06', 'date_p
osted_2018-05-07', 'date_posted_2018-05-08', 'date_posted_2018-05-09', 'date_posted_2018-05-10', 'date_posted_2018-05
-11', 'date_posted_2018-05-12', 'date_posted_2018-05-13', 'date_posted_2018-05-14', 'date_posted_2018-05-15', 'date_p
osted_2018-05-16', 'date_posted_2018-05-17', 'date_posted_2018-05-18', 'date_posted_2018-05-19', 'date_posted_2018-05
-20', 'date_posted_2018-05-21', 'date_posted_2018-05-22', 'date_posted_2018-05-23', 'date_posted_2018-05-24', 'date_p
osted_2018-05-25', 'date_posted_2018-05-26', 'date_posted_2018-05-27', 'date_posted_2018-05-28', 'date_posted_2018-05
-29', 'date_posted_2018-05-30', 'date_posted_2018-05-31', 'date_posted_2018-06-01', 'date_posted_2018-06-02', 'date_p
osted_2018-06-03', 'date_posted_2018-06-04', 'date_posted_2018-06-05', 'date_posted_2018-06-06', 'date_posted_2018-06
-07', 'date_posted_2018-06-08', 'date_posted_2018-06-09', 'date_posted_2018-06-10', 'date_posted_2018-06-11', 'date_p
osted_2018-06-12', 'date_posted_2018-06-13', 'date_posted_2018-06-14', 'date_posted_2018-06-15', 'date_posted_2018-06
-16', 'date_posted_2018-06-17', 'date_posted_2018-06-18', 'date_posted_2018-06-19', 'date_posted_2018-06-20', 'date_p
osted_2018-06-21', 'date_posted_2018-06-22', 'date_posted_2018-06-23', 'date_posted_2018-06-24', 'date_posted_2018-06
-25', 'date_posted_2018-06-26', 'date_posted_2018-06-27', 'date_posted_2018-06-28', 'date_posted_2018-06-29', 'date_p
osted_2018-06-30', 'date_posted_2018-07-01', 'date_posted_2018-07-02', 'date_posted_2018-07-03', 'date_posted_2018-07
-04', 'date_posted_2018-07-05', 'date_posted_2018-07-06', 'date_posted_2018-07-07', 'date_posted_2018-07-08', 'date_p
osted_2018-07-09', 'date_posted_2018-07-10', 'date_posted_2018-07-11', 'date_posted_2018-07-12', 'date_posted_2018-07
-13', 'date_posted_2018-07-14', 'date_posted_2018-07-15', 'date_posted_2018-07-16', 'date_posted_2018-07-17', 'date_p
```

Figure 2.13: logistic regression 5000

```
['model_year', 'cylinders', 'odometer', 'days_listed']
['condition_fair', 'condition_good', 'condition_like new', 'condition_new', 'condition_salvage', 'date_posted_2018-05
-02', 'date_posted_2018-05-03', 'date_posted_2018-05-04', 'date_posted_2018-05-05', 'date_posted_2018-05-06', 'date_p
osted_2018-05-07', 'date_posted_2018-05-08', 'date_posted_2018-05-09', 'date_posted_2018-05-10', 'date_posted_2018-05
-11', 'date_posted_2018-05-12', 'date_posted_2018-05-13', 'date_posted_2018-05-14', 'date_posted_2018-05-15', 'date_p
osted_2018-05-16', 'date_posted_2018-05-17', 'date_posted_2018-05-18', 'date_posted_2018-05-19', 'date_posted_2018-05
-20', 'date_posted_2018-05-21', 'date_posted_2018-05-22', 'date_posted_2018-05-23', 'date_posted_2018-05-24', 'date_p
osted_2018-05-25', 'date_posted_2018-05-26', 'date_posted_2018-05-27', 'date_posted_2018-05-28', 'date_posted_2018-05
-29', 'date_posted_2018-05-30', 'date_posted_2018-05-31', 'date_posted_2018-06-01', 'date_posted_2018-06-02', 'date_p
osted_2018-06-03', 'date_posted_2018-06-04', 'date_posted_2018-06-05', 'date_posted_2018-06-06', 'date_posted_2018-06
-07', 'date_posted_2018-06-08', 'date_posted_2018-06-09', 'date_posted_2018-06-10', 'date_posted_2018-06-11', 'date_p
osted_2018-06-12', 'date_posted_2018-06-13', 'date_posted_2018-06-14', 'date_posted_2018-06-15', 'date_posted_2018-06
-16', 'date_posted_2018-06-17', 'date_posted_2018-06-18', 'date_posted_2018-06-19', 'date_posted_2018-06-20', 'date_p
osted_2018-06-21', 'date_posted_2018-06-22', 'date_posted_2018-06-23', 'date_posted_2018-06-24', 'date_posted_2018-06
-25', 'date_posted_2018-06-26', 'date_posted_2018-06-27', 'date_posted_2018-06-28', 'date_posted_2018-06-29', 'date_p
osted_2018-06-30', 'date_posted_2018-07-01', 'date_posted_2018-07-02', 'date_posted_2018-07-03', 'date_posted_2018-07
-04', 'date_posted_2018-07-05', 'date_posted_2018-07-06', 'date_posted_2018-07-07', 'date_posted_2018-07-08', 'date_p
osted_2018-07-09', 'date_posted_2018-07-10', 'date_posted_2018-07-11', 'date_posted_2018-07-12', 'date_posted_2018-07
-13', 'date_posted_2018-07-14', 'date_posted_2018-07-15', 'date_posted_2018-07-16', 'date_posted_2018-07-17', 'date_p
```

Figure 2.14: logistic regression 25000

```
(23932, 490)
(5984, 490)

1        0.015168
6995     0.014040
5995     0.013914
3500     0.012243
4500     0.012076
           ...
10497    0.000042
39777    0.000042
37888    0.000042
14498    0.000042
22400    0.000042
Name: target, Length: 2451, dtype: float64

1        0.017045
6995     0.014706
5500     0.012868
7995     0.012701
5995     0.012199
           ...
23090    0.000167
12980    0.000167
18910    0.000167
17545    0.000167
27795    0.000167
Name: target, Length: 1210, dtype: float64
```

Figure 2.15: logistic regression 50000

```
Accuracy =  0.05930807248764415
Precision =  0.05930807248764415
Recall =  0.05930807248764415
F1 score =  0.05930807248764415

Classification Report
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         1
         111       0.00      0.00      0.00         1
         176       0.00      0.00      0.00         1
         196       0.00      0.00      0.00         1
         289       0.00      0.00      0.00         0
         295       1.00      1.00      1.00         1
         600       0.00      0.00      0.00         1
         850       0.00      0.00      0.00         1
         900       0.00      0.00      0.00         1
        1000       0.00      0.00      0.00         2
        1200       0.00      0.00      0.00         2
```

Figure 2.16: logistic regression 5000 accuracy

```
Accuracy =  0.058419243986254296
Precision =  0.058419243986254296
Recall =  0.058419243986254296
F1 score =  0.058419243986254296

Classification Report
              precision    recall  f1-score   support

           1       0.36      0.48      0.41        73
          69       0.00      0.00      0.00         2
         196       0.00      0.00      0.00         1
         245       0.00      0.00      0.00         1
         247       0.00      0.00      0.00         1
         267       0.00      0.00      0.00         1
         276       0.00      0.00      0.00         1
         295       0.00      0.00      0.00         2
         340       0.00      0.00      0.00         1
         351       0.00      0.00      0.00         1
         372       0.00      0.00      0.00         1
```

Figure 2.17: logistic regression 25000 accuracy

```
Accuracy =  0.05765374331550802
Precision =  0.05765374331550802
Recall =  0.05765374331550802
F1 score =  0.05765374331550802

Classification Report
              precision    recall  f1-score   support

           1       0.35      0.47      0.40       102
          24       0.00      0.00      0.00         1
          69       0.00      0.00      0.00         1
         111       0.00      0.00      0.00         1
         180       0.00      0.00      0.00         1
         195       0.00      0.00      0.00         1
         196       0.00      0.00      0.00         1
         199       0.00      0.00      0.00         1
         200       0.00      0.00      0.00         1
         211       0.00      0.00      0.00         2
         237       0.00      0.00      0.00         1
```

Figure 2.18: logistic regression 50000 accuracy

3. LINEAR REGRESSION

The next method I tried was linear regression. Linear Regression is a supervised machine learning algorithm that is used for predicting a continuous target variable based on one or more input features. It is a linear approach to modeling the relationship between a scalar dependent variable y and one or more independent variables denoted by X.

The accuracy was the best with the 5000 row dataset with linear regression and it was 50000 rows and 25000 rows next.

I used dropping the rows or columns method with this regression as well and it increased accuracy. After that I removed the duplicated rows and duplicate values, and converted some strings and hard to manipulate values into numerical values and used ordinal encoder. These were my preprocessing steps.

The followings are the evaluation of the preprocessing steps.

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 3.1: linear regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
5092	17999	2014.0	ram 1500	like new	8.0	gas	154000.0	automatic	pickup	white	1.0	2018-06-21	11
5093	4800	2012.0	volkswagen jetta	good	4.0	gas	138000.0	automatic	sedan	silver	NaN	2018-07-19	39
5094	12000	2005.0	chevrolet silverado 2500hd	good	8.0	diesel	228000.0	automatic	pickup	silver	1.0	2018-08-18	52
5095	8999	2011.0	jeep grand cherokee	good	6.0	gas	NaN	automatic	SUV	white	1.0	2018-11-01	51
5096	11000	2012.0	dodge charger	excellent	8.0	gas	81000.0	automatic	sedan	black	NaN	2019-04-19	44

Figure 3.2: linear regression 5000

After these steps, I used ordinal encoding. Ordinal Encoding is a technique

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	59995	NaN	ram 2500	like new	6.0	diesel	22635.0	automatic	truck	white	1.0	2018-06-02	52
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	61030.0	automatic	truck	black	1.0	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	17037.0	automatic	bus	white	NaN	2018-08-20	73
3	2850	1997.0	ford f150	good	8.0	gas	255040.0	manual	pickup	NaN	1.0	2018-06-05	18
4	3700	2007.0	honda accord	fair	NaN	gas	173600.0	manual	sedan	red	NaN	2018-10-02	26

Figure 3.3: linear regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
25139	12000	2004.0	jeep wrangler	excellent	6.0	gas	130000.0	automatic	offroad	NaN	1.0	2018-10-15	8
25140	25200	2014.0	chevrolet silverado 1500 crew	good	6.0	gas	85048.0	automatic	pickup	silver	1.0	2019-04-06	54
25141	2650	2006.0	toyota sienna	good	6.0	gas	202000.0	automatic	mini-van	red	NaN	2018-11-24	47
25142	12999	2012.0	chevrolet colorado	good	4.0	gas	92816.0	automatic	truck	NaN	1.0	2018-09-12	61
25143	1000	1997.0	ford expedition	fair	8.0	gas	NaN	automatic	SUV	NaN	1.0	2018-06-01	113

Figure 3.4: linear regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 3.5: linear regression 50000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
51520	9249	2013.0	nissan maxima	like new	6.0	gas	88136.0	automatic	sedan	black	NaN	2018-10-03	37
51521	2700	2002.0	honda civic	salvage	4.0	gas	181500.0	automatic	sedan	white	NaN	2018-11-14	22
51522	3950	2009.0	hyundai sonata	excellent	4.0	gas	128000.0	automatic	sedan	blue	NaN	2018-11-15	32
51523	7455	2013.0	toyota corolla	good	4.0	gas	139573.0	automatic	sedan	black	NaN	2018-07-02	71
51524	6300	2014.0	nissan altima	good	4.0	gas	NaN	automatic	sedan	NaN	NaN	2018-06-05	10

Figure 3.6: linear regression 50000

```
Int64Index: 5097 entries, 0 to 5096
Data columns (total 13 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          5097 non-null   int64
 1   model_year     4740 non-null   float64
 2   model          5097 non-null   object
 3   condition      5097 non-null   object
 4   cylinders      4603 non-null   float64
 5   fuel           5097 non-null   object
 6   odometer       4354 non-null   float64
 7   transmission   5097 non-null   object
 8   type           5097 non-null   object
 9   paint_color    4194 non-null   object
 10  is_4wd         2655 non-null   float64
 11  date_posted    5097 non-null   object
 12  days_listed    5097 non-null   int64
dtypes: float64(4), int64(2), object(7)
memory usage: 557.5+ KB
```

Figure 3.7: linear regression 5000

```
Data columns (total 13 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          25144 non-null  int64
 1   model_year     23345 non-null  float64
 2   model          25144 non-null  object
 3   condition      25144 non-null  object
 4   cylinders      22585 non-null  float64
 5   fuel           25144 non-null  object
 6   odometer       21346 non-null  float64
 7   transmission   25144 non-null  object
 8   type           25144 non-null  object
 9   paint_color    20587 non-null  object
 10  is_4wd         12352 non-null  float64
 11  date_posted    25144 non-null  object
 12  days_listed    25144 non-null  int64
dtypes: float64(4), int64(2), object(7)
memory usage: 2.7+ MB
```

Figure 3.8: linear regression 25000

```
Int64Index: 51525 entries, 0 to 51524
Data columns (total 13 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          51525 non-null  int64
 1   model_year     47906 non-null  float64
 2   model          51525 non-null  object
 3   condition      51525 non-null  object
 4   cylinders      46265 non-null  float64
 5   fuel           51525 non-null  object
 6   odometer       43633 non-null  float64
 7   transmission   51525 non-null  object
 8   type           51525 non-null  object
 9   paint_color    42258 non-null  object
 10  is_4wd         25572 non-null  float64
 11  date_posted    51525 non-null  object
 12  days_listed    51525 non-null  int64
dtypes: float64(4), int64(2), object(7)
memory usage: 5.5+ MB
```

Figure 3.9: linear regression 50000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	1450000	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	887050	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	1100000	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas		automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	809030	automatic	sedan	black	NaN	2019-04-02	28

Figure 3.10: linear regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	59995	NaN	ram 2500	like new	6.0	diesel	226350	automatic	truck	white	1.0	2018-06-02	52
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	610300	automatic	truck	black	1.0	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	170370	automatic	bus	white	NaN	2018-08-20	73
3	2850	1997.0	ford f150	good	8.0	gas	2550400	manual	pickup	NaN	1.0	2018-06-05	18
4	3700	2007.0	honda accord	fair	NaN	gas	1736000	manual	sedan	red	NaN	2018-10-02	26

Figure 3.11: linear regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	1450000	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	887050	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	1100000	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas		automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	809030	automatic	sedan	black	NaN	2019-04-02	28

Figure 3.12: linear regression 50000

```
price              0
model_year       357
model              0
condition          0
cylinders        494
fuel               0
odometer         743
transmission       0
type               0
paint_color      903
is_4wd          2442
date_posted        0
days_listed        0
dtype: int64
```

Figure 3.13: linear regression 5000

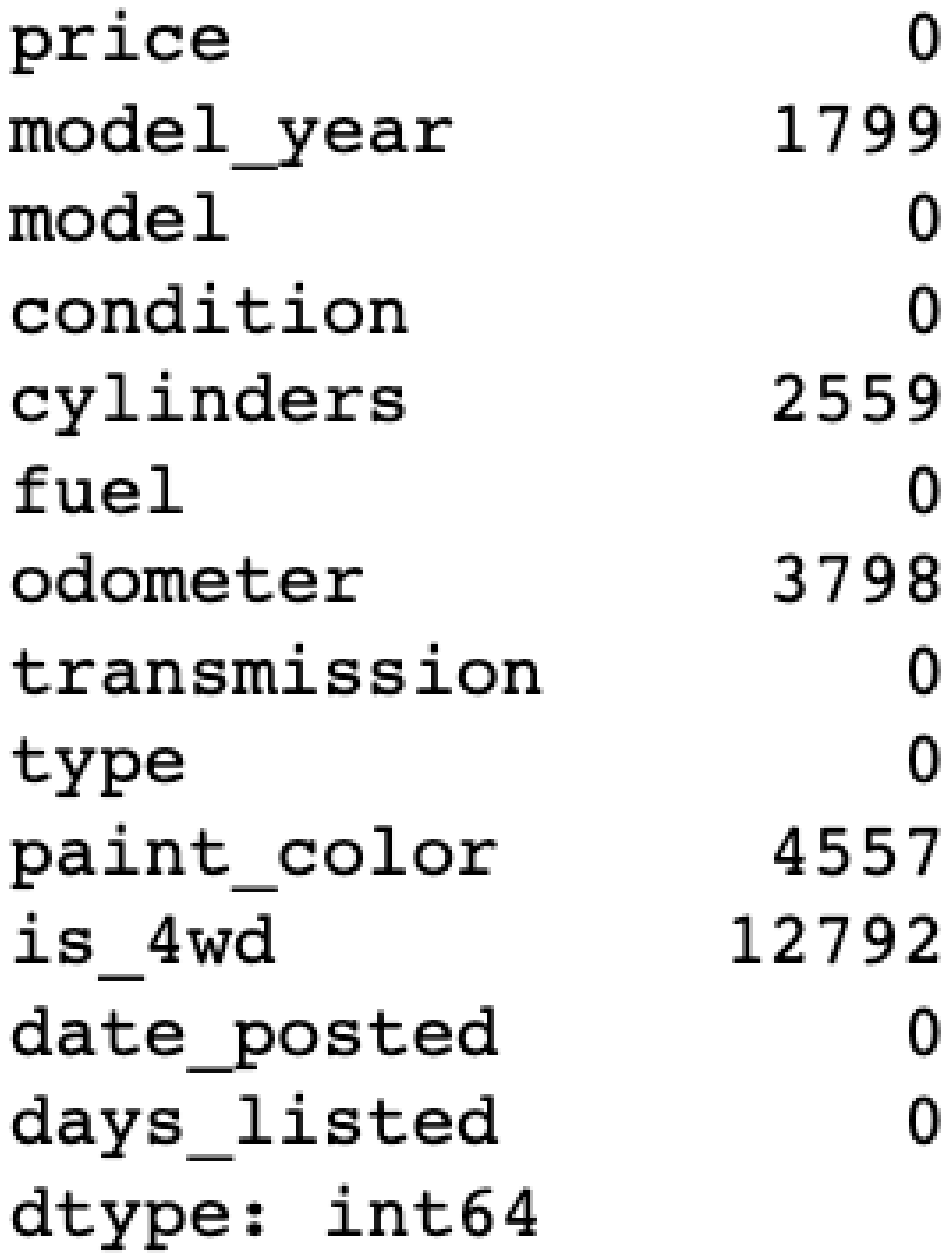

Figure 3.14: linear regression 25000

```
price             0
model_year     3619
model             0
condition         0
cylinders      5260
fuel              0
odometer       7892
transmission      0
type              0
paint_color    9267
is_4wd        25953
date_posted       0
days_listed       0
dtype: int64
```

Figure 3.15: linear regression 50000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
2	5500	2013.0	hyundai sonata	like new	4.0	gas	1100000.0	automatic	sedan	red	2019-02-07	79
4	14900	2017.0	chrysler 200	excellent	4.0	gas	809030.0	automatic	sedan	black	2019-04-02	28
5	14990	2014.0	chrysler 300	excellent	6.0	gas	579540.0	automatic	sedan	black	2018-06-20	15
6	12990	2015.0	toyota camry	excellent	4.0	gas	792120.0	automatic	sedan	white	2018-12-27	73
7	15990	2013.0	honda pilot	excellent	6.0	gas	1094730.0	automatic	SUV	black	2019-01-07	68

Figure 3.16: linear regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	610300.0	automatic	truck	black	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	170370.0	automatic	bus	white	2018-08-20	73
5	7000	2002.0	toyota tacoma	fair	6.0	gas	1282000.0	automatic	truck	silver	2018-08-09	24
6	3900	2003.0	jeep liberty	good	4.0	gas	1288720.0	automatic	SUV	brown	2018-06-26	40
10	8000	2008.0	ford f-250 super duty	good	10.0	gas	2400000.0	manual	truck	red	2019-04-13	49

Figure 3.17: linear regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	date_posted	days_listed
2	5500	2013.0	hyundai sonata	like new	4.0	gas	1100000.0	automatic	sedan	red	2019-02-07	79
4	14900	2017.0	chrysler 200	excellent	4.0	gas	809030.0	automatic	sedan	black	2019-04-02	28
5	14990	2014.0	chrysler 300	excellent	6.0	gas	579540.0	automatic	sedan	black	2018-06-20	15
6	12990	2015.0	toyota camry	excellent	4.0	gas	792120.0	automatic	sedan	white	2018-12-27	73
7	15990	2013.0	honda pilot	excellent	6.0	gas	1094730.0	automatic	SUV	black	2019-01-07	68

Figure 3.18: linear regression 50000

used to convert categorical variables that have an inherent order or ranking to numerical values. It is used to represent ordinal categorical variables, which are categorical variables that have a natural ordering, such as ”low”, ”medium”, ”high”.

The basic idea behind ordinal encoding is to map each category to a unique integer value, such that the categories are assigned values that reflect their relative order. For example, in a variable with the categories ”low”, ”medium”, ”high”, ”low” would be encoded as 0, ”medium” as 1 and ”high” as 2.

Followings are the correlation matrices for the datasets.

Followings are the accuracies for the three datasets.

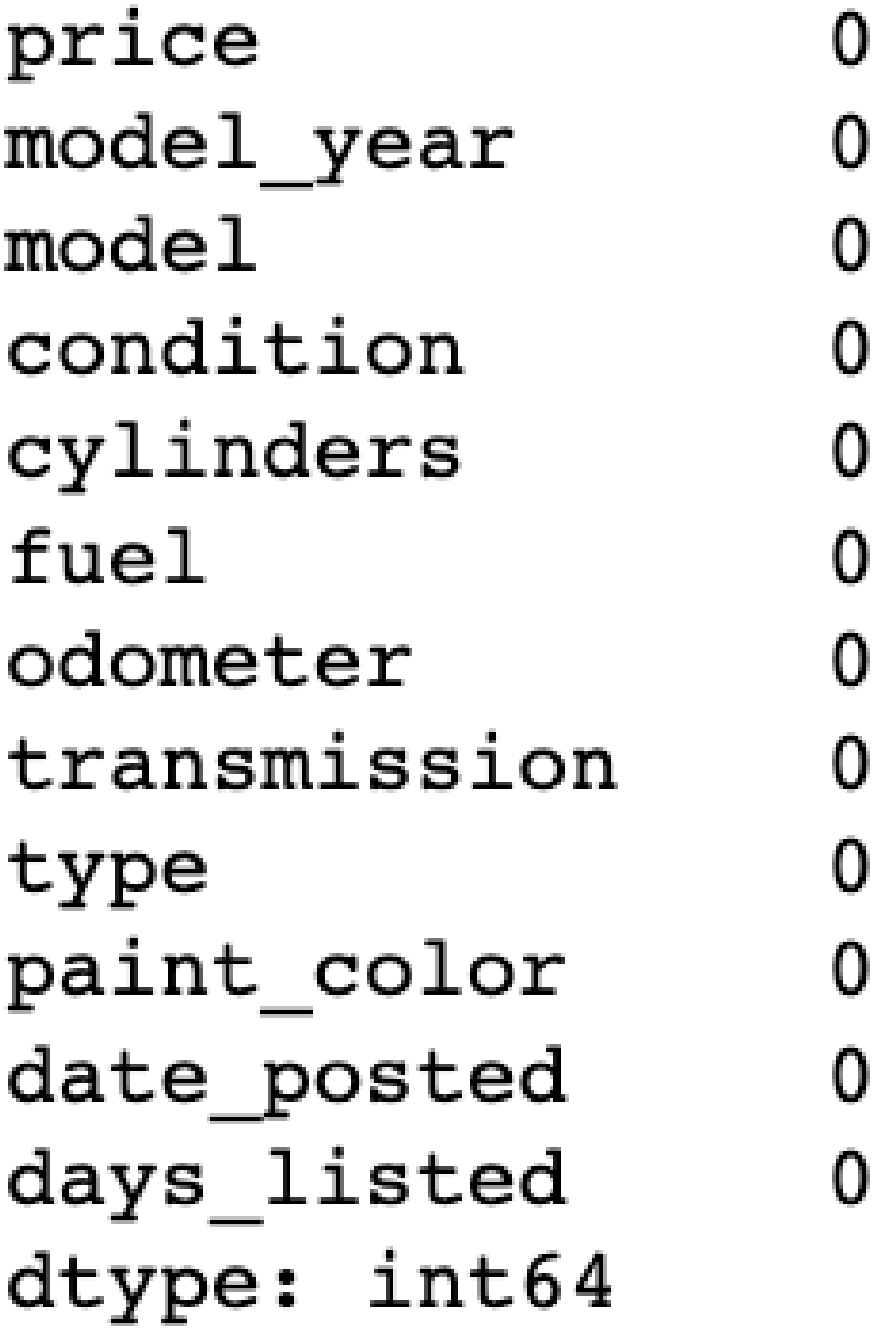

Figure 3.19: linear regression 5000

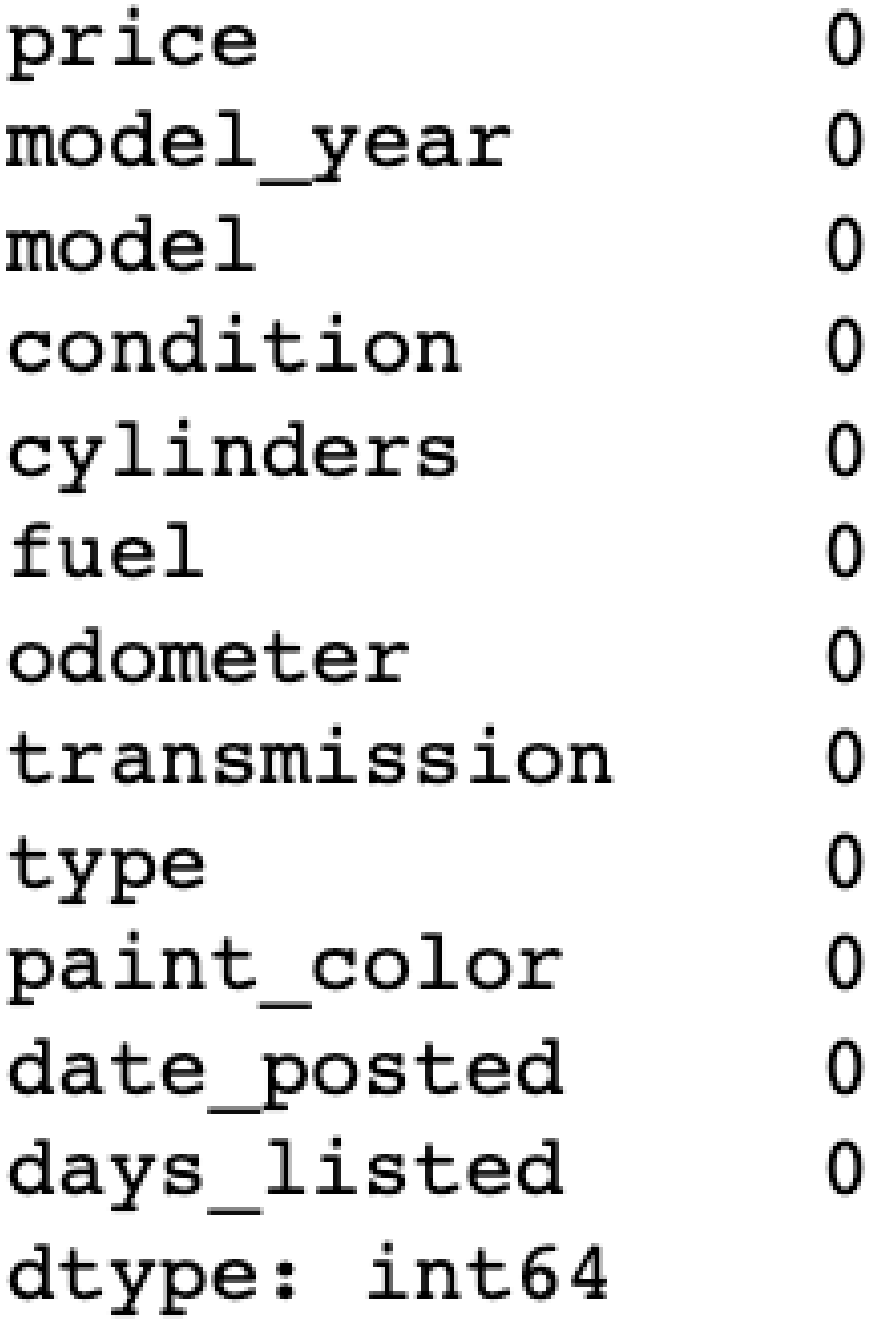

Figure 3.20: linear regression 25000

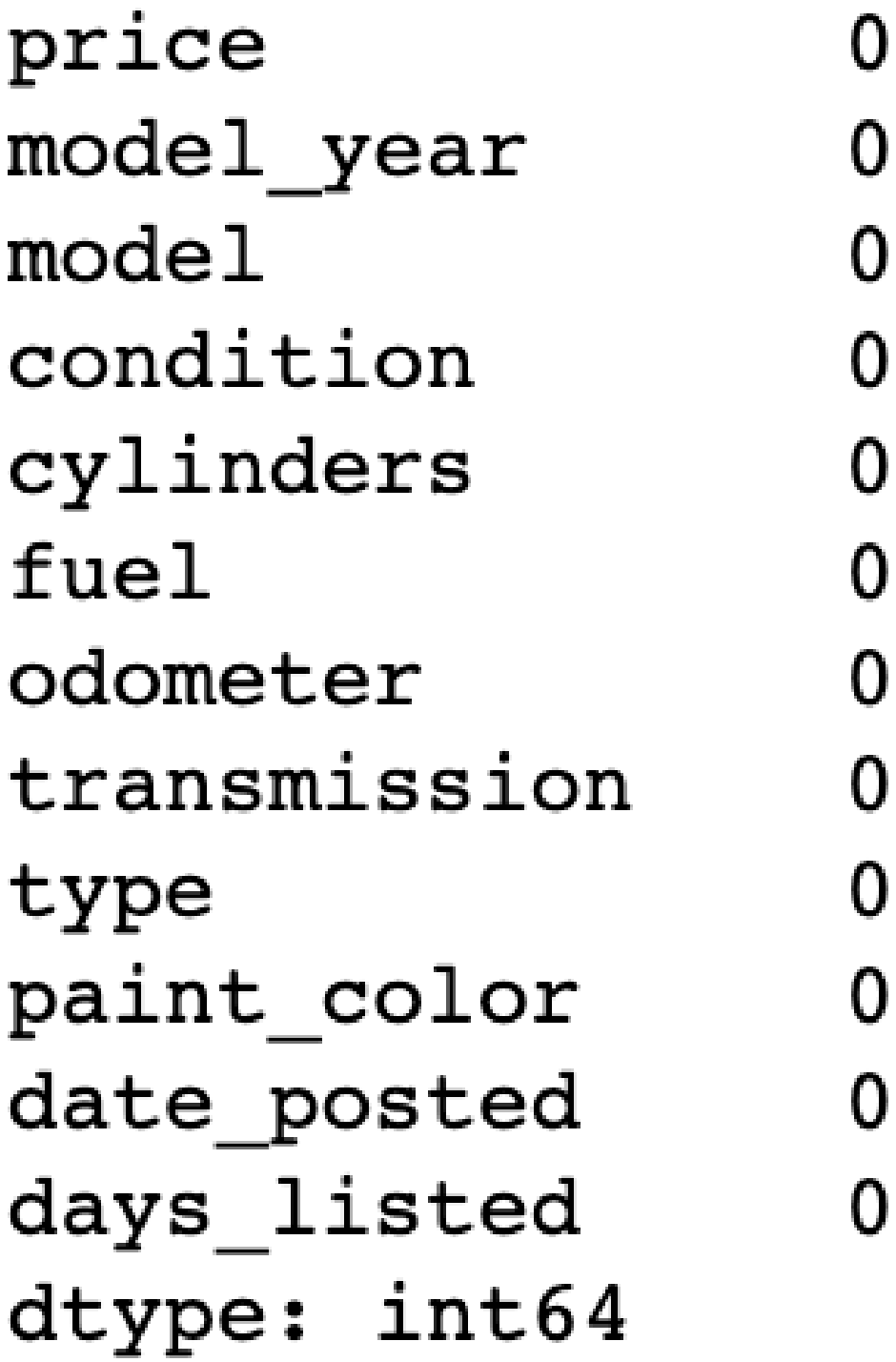

Figure 3.21: linear regression 50000

model	price	model_year	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
chrysler 300	14990	2014.0	0.0	6.0	1.0	57954.0	0.0	7.0	0.0	1.0	49.0	15
honda pilot	15990	2013.0	0.0	6.0	1.0	109473.0	0.0	0.0	0.0	1.0	247.0	68
chevrolet silverado 1500	19500	2011.0	0.0	8.0	1.0	128413.0	0.0	6.0	0.0	1.0	136.0	38
gmc yukon	12990	2009.0	0.0	8.0	1.0	132285.0	0.0	0.0	0.0	1.0	270.0	24
ram 1500	14990	2010.0	0.0	8.0	1.0	130725.0	0.0	6.0	8.0	1.0	239.0	13

Figure 3.22: linear regression 5000

model	price	model_year	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
gmc sierra 1500	15990	2001.0	2.0	8.0	2.0	61030.0	0.0	9.0	0.0	1.0	49.0	13
toyota tacoma	7000	2002.0	1.0	6.0	2.0	128200.0	0.0	9.0	9.0	1.0	100.0	24
toyota tacoma	27475	2015.0	0.0	6.0	2.0	54300.0	0.0	9.0	5.0	1.0	93.0	23
chevrolet suburban	4995	2004.0	2.0	8.0	2.0	120455.0	0.0	11.0	3.0	1.0	276.0	54
nissan rogue	17995	2017.0	0.0	4.0	2.0	37500.0	0.0	0.0	5.0	1.0	82.0	35

Figure 3.23: linear regression 25000

model	price	model_year	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
chrysler 300	14990	2014.0	0.0	6.0	2.0	57954.0	0.0	8.0	0.0	1.0	50.0	15
honda pilot	15990	2013.0	0.0	6.0	2.0	109473.0	0.0	0.0	0.0	1.0	251.0	68
chevrolet silverado 1500	19500	2011.0	0.0	8.0	2.0	128413.0	0.0	7.0	0.0	1.0	139.0	38
gmc yukon	12990	2009.0	0.0	8.0	2.0	132285.0	0.0	0.0	0.0	1.0	275.0	24
ram 1500	14990	2010.0	0.0	8.0	2.0	130725.0	0.0	7.0	8.0	1.0	243.0	13

Figure 3.24: linear regression 50000

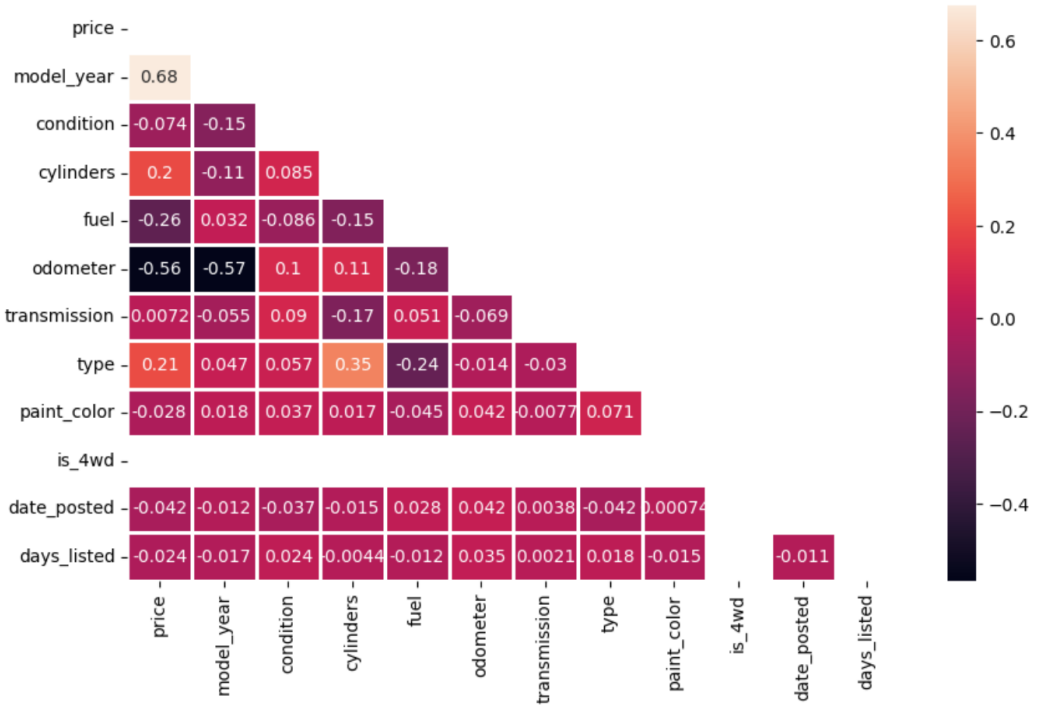

Figure 3.25: linear regression 5000

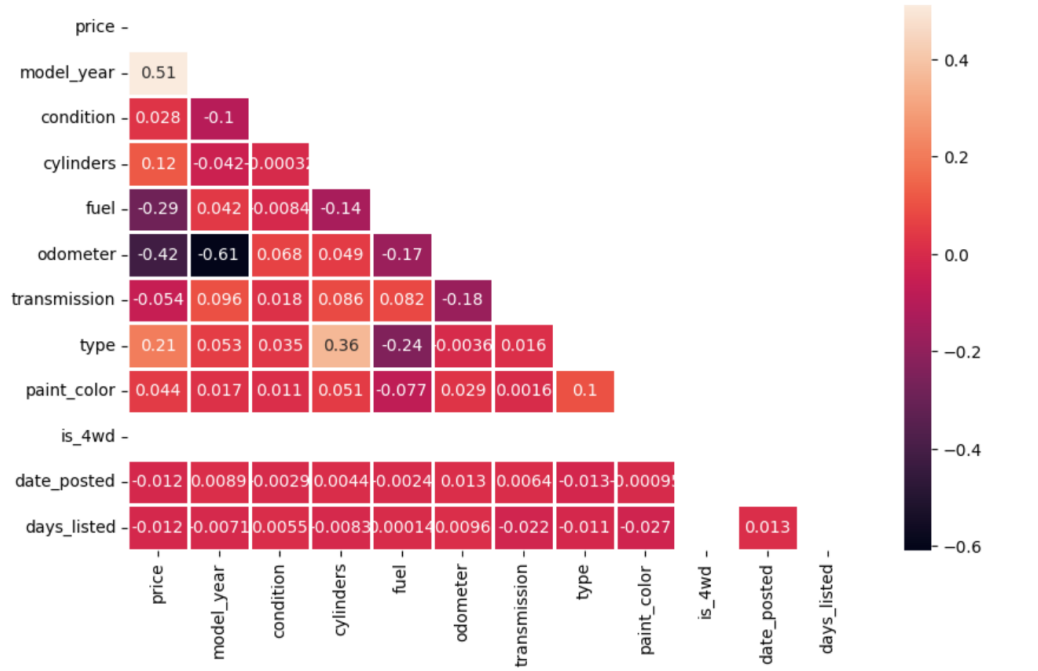

Figure 3.26: linear regression 25000

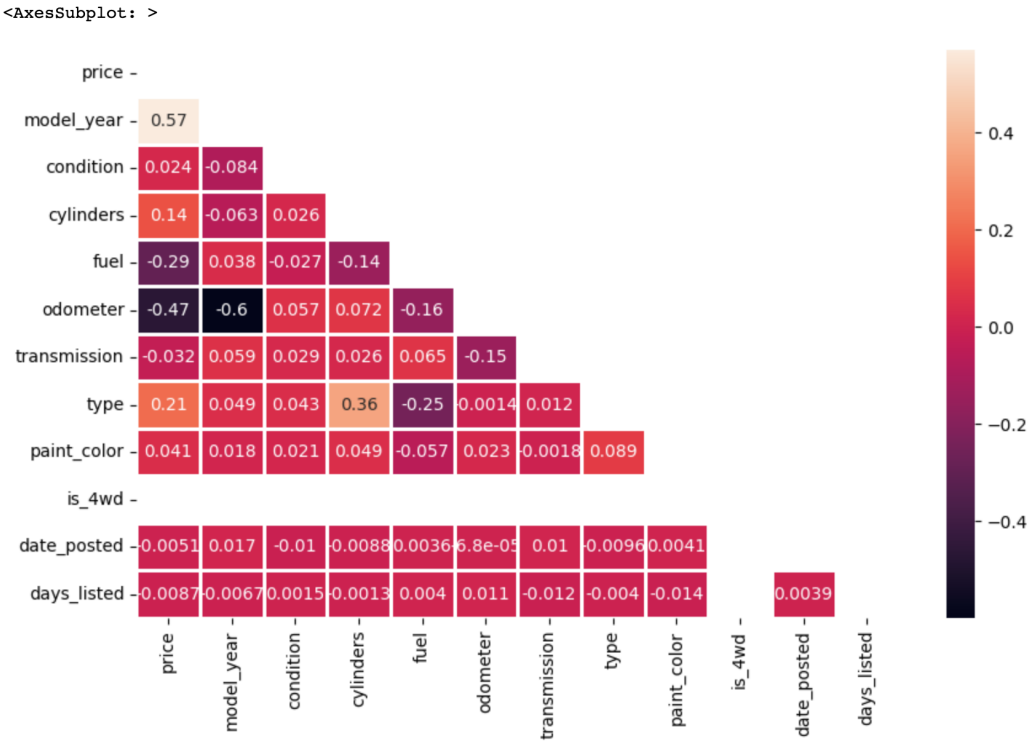

Figure 3.27: linear regression 50000

```
Score with linear regression : 0.6755086942711386
```

Figure 3.28: linear regression 5000

```
Score with linear regression : 0.430439736763793
```

Figure 3.29: linear regression 25000

```
Score with linear regression : 0.5076168397209266
```

Figure 3.30: linear regression 50000

4. RANDOM FOREST REGRESSION

The next method I tried was random forest regression. Random Forest Regression is a supervised machine learning algorithm that is used for predicting a continuous target variable based on one or more input features. It is an ensemble technique that combines the predictions of multiple decision trees to improve the overall accuracy and stability of the model.

The basic idea behind random forest regression is to train multiple decision trees on different subsets of the data, and then average the predictions of all the trees to make the final prediction. Each decision tree is trained on a random subset of the data and a random subset of the features. This randomness helps to reduce overfitting and improve the generalization of the model.

I used dropping the rows or columns method with this regression as well and it increased accuracy. After that I removed the duplicated rows and duplicate values, and converted some strings and hard to manipulate values into numerical values. These were my preprocessing steps.

The followings are the evaluation of the preprocessing, training and testing steps.

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 4.1: linear regression 5000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	59995	NaN	ram 2500	like new	6.0	diesel	22635.0	automatic	truck	white	1.0	2018-06-02	52
1	15990	2001.0	gmc sierra 1500	good	8.0	gas	61030.0	automatic	truck	black	1.0	2018-06-19	13
2	28990	2011.0	ford econoline	good	10.0	gas	17037.0	automatic	bus	white	NaN	2018-08-20	73
3	2850	1997.0	ford f150	good	8.0	gas	255040.0	manual	pickup	NaN	1.0	2018-06-05	18
4	3700	2007.0	honda accord	fair	NaN	gas	173600.0	manual	sedan	red	NaN	2018-10-02	26

Figure 4.2: linear regression 25000

	price	model_year	model	condition	cylinders	fuel	odometer	transmission	type	paint_color	is_4wd	date_posted	days_listed
0	9400	2011.0	bmw x5	good	6.0	gas	145000.0	automatic	SUV	NaN	1.0	2018-06-23	19
1	25500	NaN	ford f-150	good	6.0	gas	88705.0	automatic	pickup	white	1.0	2018-10-19	50
2	5500	2013.0	hyundai sonata	like new	4.0	gas	110000.0	automatic	sedan	red	NaN	2019-02-07	79
3	1500	2003.0	ford f-150	fair	8.0	gas	NaN	automatic	pickup	NaN	NaN	2019-03-22	9
4	14900	2017.0	chrysler 200	excellent	4.0	gas	80903.0	automatic	sedan	black	NaN	2019-04-02	28

Figure 4.3: linear regression 50000

```
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          5097 non-null   int64
 1   model_year     4740 non-null   float64
 2   model          5097 non-null   object
 3   condition      5097 non-null   object
 4   cylinders      4603 non-null   float64
 5   fuel           5097 non-null   object
 6   odometer       4354 non-null   float64
 7   transmission   5097 non-null   object
 8   type           5097 non-null   object
 9   paint_color    4194 non-null   object
 10  date_posted    5097 non-null   object
 11  days_listed    5097 non-null   int64
dtypes: float64(3), int64(2), object(7)
memory usage: 478.0+ KB
```

Figure 4.4: linear regression 5000

```
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          25144 non-null  int64
 1   model_year     23345 non-null  float64
 2   model          25144 non-null  object
 3   condition      25144 non-null  object
 4   cylinders      22585 non-null  float64
 5   fuel           25144 non-null  object
 6   odometer       21346 non-null  float64
 7   transmission   25144 non-null  object
 8   type           25144 non-null  object
 9   paint_color    20587 non-null  object
 10  date_posted    25144 non-null  object
 11  days_listed    25144 non-null  int64
dtypes: float64(3), int64(2), object(7)
memory usage: 2.3+ MB
```

Figure 4.5: linear regression 25000

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51525 entries, 0 to 51524
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   price         51525 non-null  int64  
 1   model_year    47906 non-null  float64
 2   model         51525 non-null  object 
 3   condition     51525 non-null  object 
 4   cylinders     46265 non-null  float64
 5   fuel          51525 non-null  object 
 6   odometer      43633 non-null  float64
 7   transmission  51525 non-null  object 
 8   type          51525 non-null  object 
 9   paint_color   42258 non-null  object 
 10  date_posted   51525 non-null  object 
 11  days_listed   51525 non-null  int64  
dtypes: float64(3), int64(2), object(7)
memory usage: 4.7+ MB
```

Figure 4.6: linear regression 50000

	price	model_year	cylinders	odometer	days_listed
count	5097.000000	4740.000000	4603.000000	4354.000000	5097.000000
mean	12776.574456	2009.737764	6.123181	114456.956362	39.823622
std	10578.789707	6.159706	1.640372	64499.434140	28.643277
min	1.000000	1963.000000	3.000000	0.000000	0.000000
25%	5500.000000	2006.000000	4.000000	70110.000000	19.000000
50%	9850.000000	2011.000000	6.000000	111964.000000	33.000000
75%	17489.000000	2014.000000	8.000000	152000.000000	54.000000
max	189000.000000	2019.000000	10.000000	866000.000000	271.000000

Figure 4.7: linear regression 5000

	price	model_year	cylinders	odometer	days_listed
count	25144.000000	23345.000000	22585.000000	21346.000000	25144.000000
mean	12010.353603	2009.886100	6.128005	114111.391877	39.375477
std	10162.241132	6.209461	1.670880	65088.674603	28.160300
min	1.000000	1948.000000	3.000000	0.000000	0.000000
25%	4995.000000	2006.000000	4.000000	68607.000000	19.000000
50%	8995.000000	2011.000000	6.000000	111963.000000	33.000000
75%	16756.750000	2014.000000	8.000000	153172.750000	53.000000
max	375000.000000	2019.000000	12.000000	990000.000000	271.000000

Figure 4.8: linear regression 25000

	price	model_year	cylinders	odometer	days_listed
count	51525.000000	47906.000000	46265.000000	43633.000000	51525.00000
mean	12132.464920	2009.750470	6.125235	115553.461738	39.55476
std	10040.803015	6.282065	1.660360	65094.611341	28.20427
min	1.000000	1908.000000	3.000000	0.000000	0.00000
25%	5000.000000	2006.000000	4.000000	70000.000000	19.00000
50%	9000.000000	2011.000000	6.000000	113000.000000	33.00000
75%	16839.000000	2014.000000	8.000000	155000.000000	53.00000
max	375000.000000	2019.000000	12.000000	990000.000000	271.00000

Figure 4.9: linear regression 50000

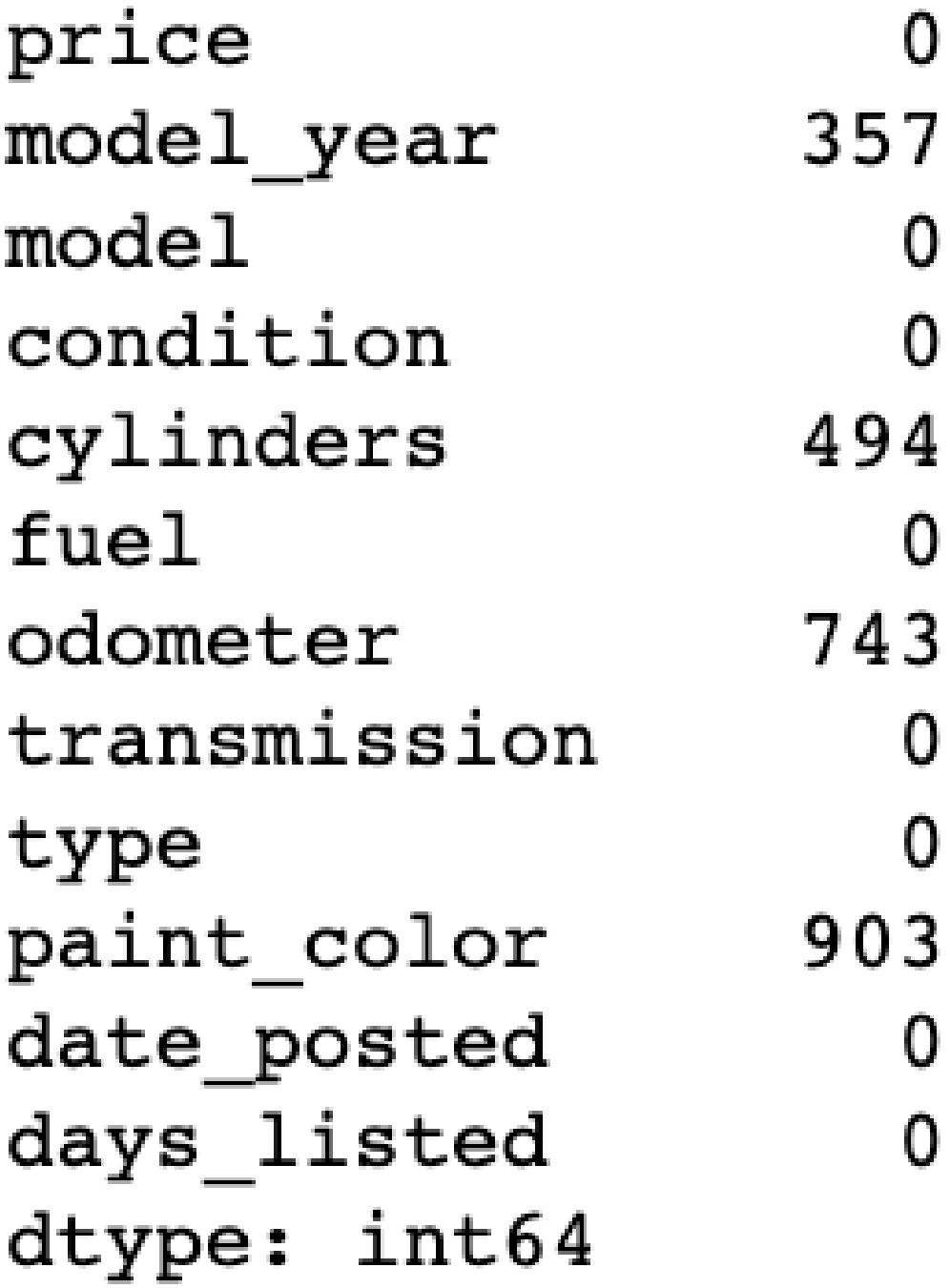

Figure 4.10: linear regression 5000

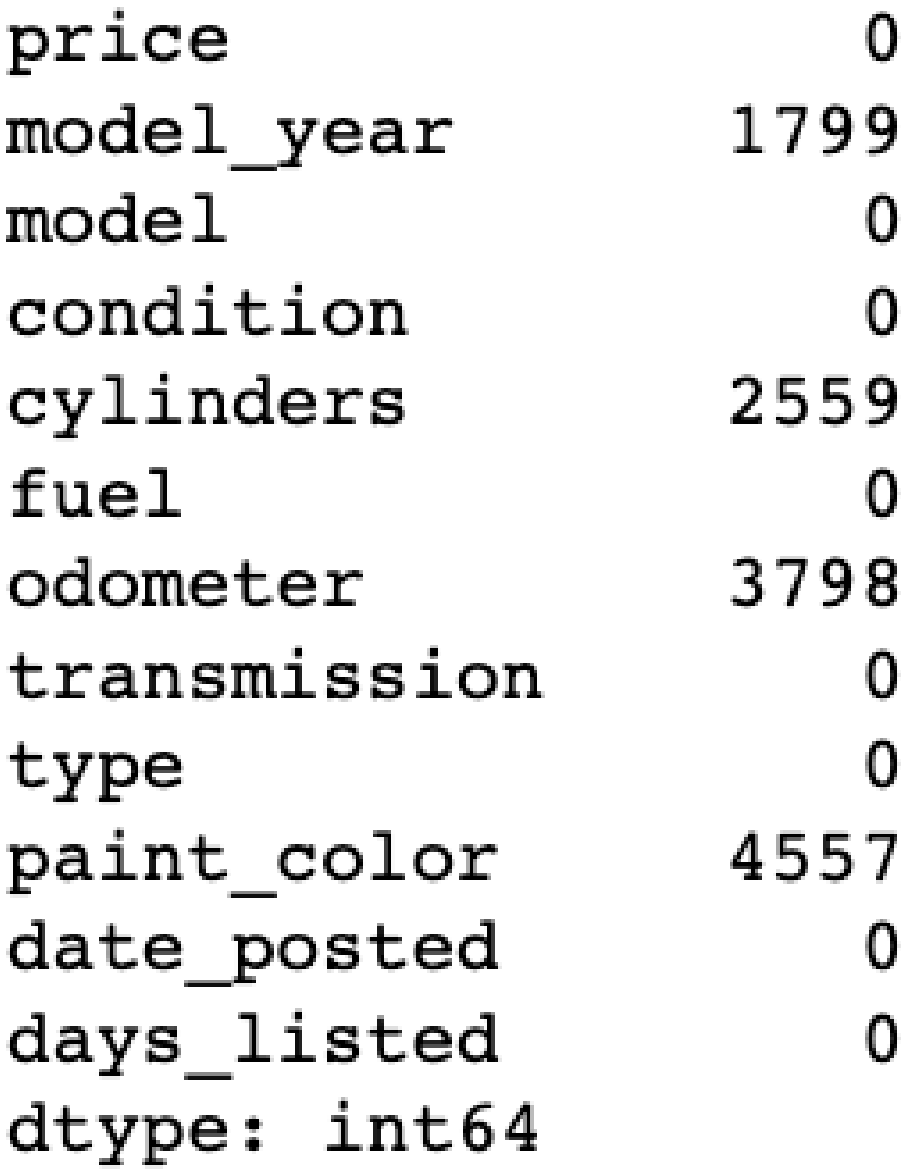

Figure 4.11: linear regression 25000

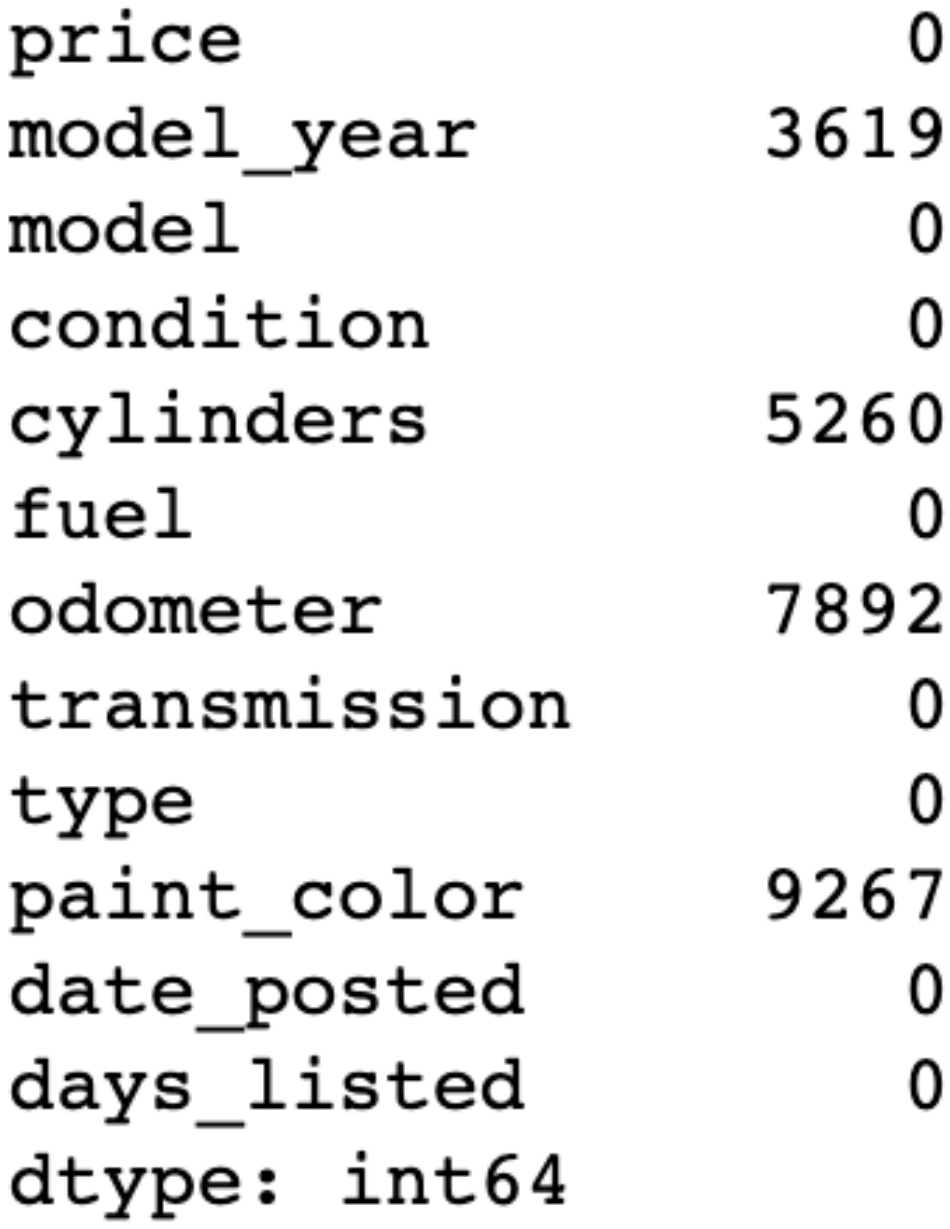

Figure 4.12: linear regression 50000

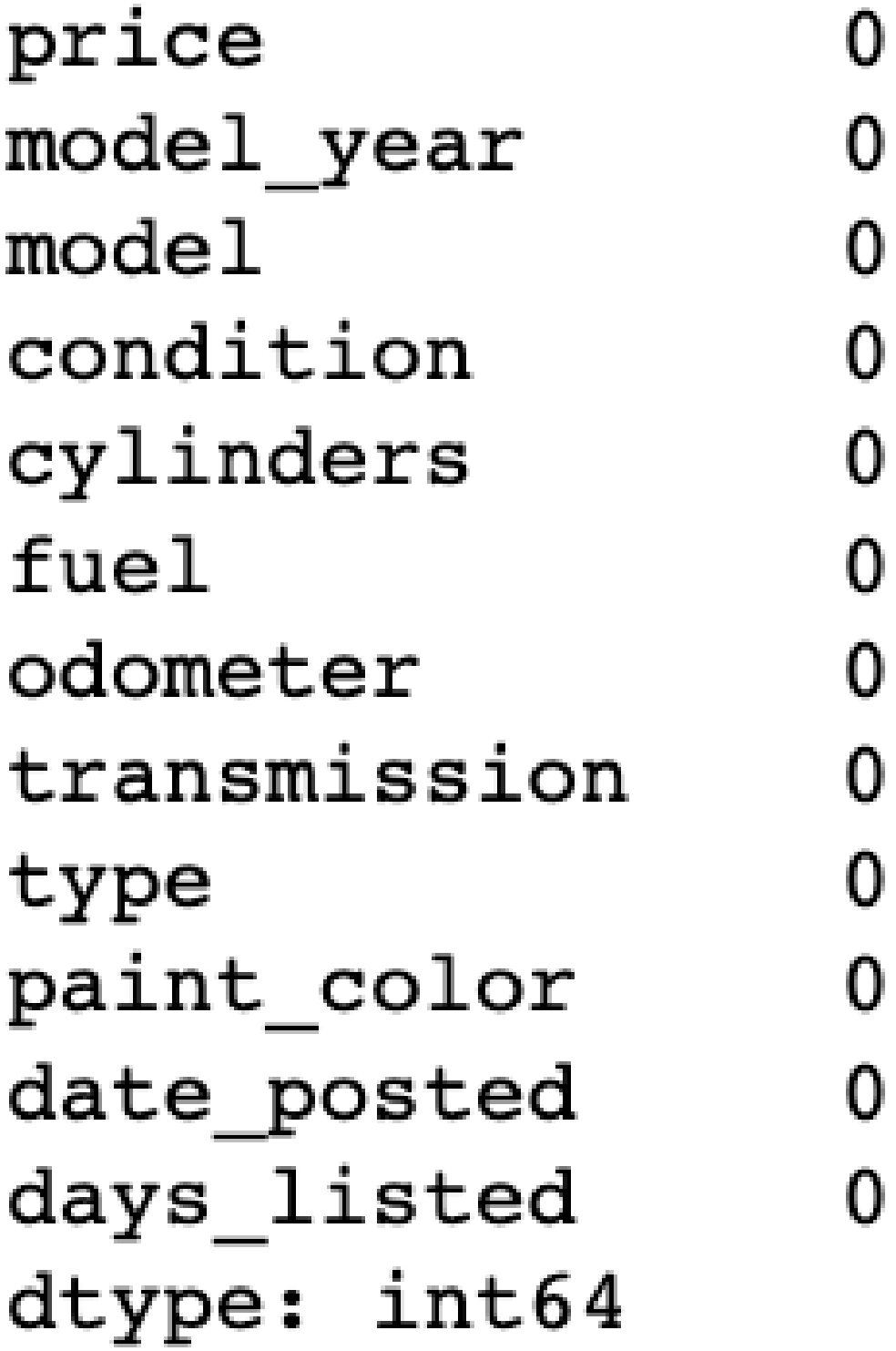

Figure 4.13: linear regression 5000

```
price           0
model_year      0
model           0
condition       0
cylinders       0
fuel            0
odometer        0
transmission    0
type            0
paint_color     0
date_posted     0
days_listed     0
dtype: int64
```

Figure 4.14: linear regression 25000

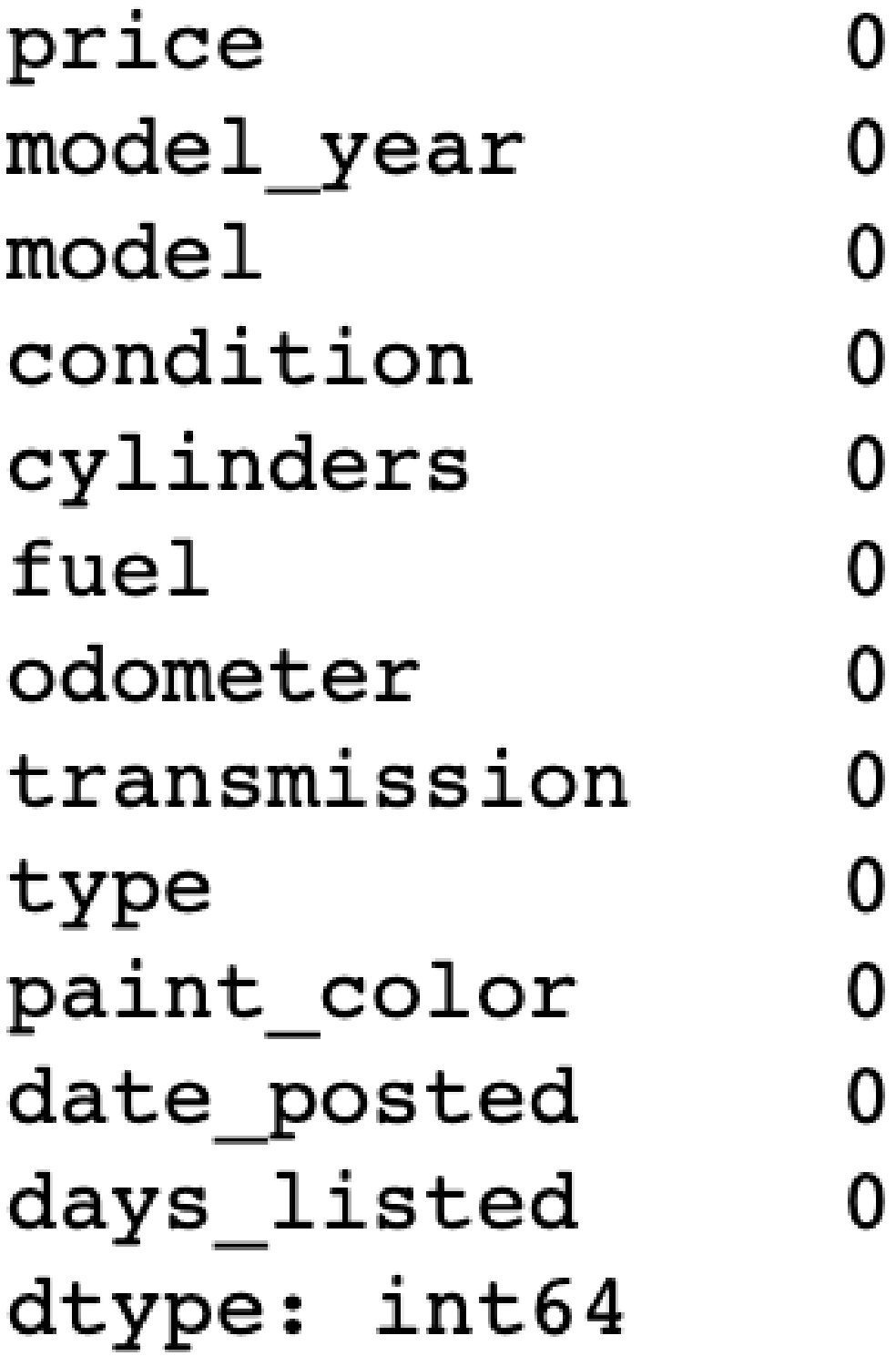

Figure 4.15: linear regression 50000

	price	model_year	cylinders	odometer	days_listed	condition_fair	condition_good	condition_like new	condition_new	condition_salvage	...	date_posted_2019-04-10
2	5500	2013.0	4.0	110000.0	79	0	0	1	0	0	...	0
4	14900	2017.0	4.0	80903.0	28	0	0	0	0	0	...	0
5	14990	2014.0	6.0	57954.0	15	0	0	0	0	0	...	0
6	12990	2015.0	4.0	79212.0	73	0	0	0	0	0	...	0
7	15990	2013.0	6.0	109473.0	68	0	0	0	0	0	...	0

5 rows × 391 columns

Figure 4.16: linear regression 5000

	price	model_year	cylinders	odometer	days_listed	condition_fair	condition_good	condition_like new	condition_new	condition_salvage	...	date_posted_2019-04-10
1	15990	2001.0	8.0	61030.0	13	0	1	0	0	0	...	0
2	28990	2011.0	10.0	17037.0	73	0	1	0	0	0	...	0
5	7000	2002.0	6.0	128200.0	24	1	0	0	0	0	...	0
6	3900	2003.0	4.0	128872.0	40	0	1	0	0	0	...	0
10	8000	2008.0	10.0	240000.0	49	0	1	0	0	0	...	0

5 rows × 392 columns

Figure 4.17: linear regression 25000

	price	model_year	cylinders	odometer	days_listed	condition_fair	condition_good	condition_like new	condition_new	condition_salvage	...	date_posted_2019-04-10
2	5500	2013.0	4.0	110000.0	79	0	0	1	0	0	...	0
4	14900	2017.0	4.0	80903.0	28	0	0	0	0	0	...	0
5	14990	2014.0	6.0	57954.0	15	0	0	0	0	0	...	0
6	12990	2015.0	4.0	79212.0	73	0	0	0	0	0	...	0
7	15990	2013.0	6.0	109473.0	68	0	0	0	0	0	...	0

5 rows × 392 columns

Figure 4.18: linear regression 50000

	price	model_year	cylinders	odometer	days_listed	condition_fair	condition_good	condition_like new	condition_new	condition_salvage	...
price	1.000000	0.462674	0.320491	-0.444373	-0.006319	-0.169789	-0.090744	0.093840	0.106407	-0.064351	...
model_year	0.462674	1.000000	-0.184523	-0.451756	-0.016731	-0.252617	-0.144323	0.094518	0.056385	-0.072680	...
cylinders	0.320491	-0.184523	1.000000	0.097961	0.015492	0.056963	0.096341	-0.030330	0.032803	0.012792	...
odometer	-0.444373	-0.451756	0.097961	1.000000	0.016006	0.158431	0.141675	-0.138779	-0.063191	0.119655	...
days_listed	-0.006319	-0.016731	0.015492	0.016006	1.000000	0.008399	-0.007324	0.019750	0.001472	-0.005184	...
...	...	...	...	...	...	...	...	...	...	...	...
date_posted_2019-04-15	-0.030435	-0.043300	0.008961	0.027595	-0.023806	-0.011656	0.009132	-0.017253	-0.003294	-0.004111	...
date_posted_2019-04-16	0.025437	0.011931	0.014784	-0.018767	-0.006385	0.031263	-0.004377	-0.012732	-0.002430	-0.003034	...
date_posted_2019-04-17	-0.023361	-0.000561	-0.028513	0.037072	-0.004747	-0.011112	0.037784	0.027092	0.102601	-0.003919	...
date_posted_2019-04-18	0.016953	0.027936	0.013197	-0.002780	-0.005438	-0.009292	0.018859	-0.013754	-0.002626	-0.003277	...
date_posted_2019-04-19	-0.031821	-0.013547	0.010773	0.008170	-0.009391	-0.010540	0.007122	0.007343	-0.002978	-0.003717	...

391 rows × 391 columns

Figure 4.19: linear regression 5000

	price	model_year	cylinders	odometer	days_listed	condition_fair	condition_good	condition_like new	condition_new	condition_salvage	...
price	1.000000	0.406099	0.283704	-0.397556	0.000368	-0.160758	-0.072517	0.122177	0.051320	-0.032680	...
model_year	0.406099	1.000000	-0.140090	-0.477682	-0.014973	-0.227708	-0.155747	0.115007	0.028539	-0.023173	...
cylinders	0.283704	-0.140090	1.000000	0.069175	0.011605	0.035859	0.052098	-0.038014	-0.006160	-0.005893	...
odometer	-0.397556	-0.477682	0.069175	1.000000	0.007802	0.181109	0.168963	-0.167369	-0.044411	0.027427	...
days_listed	0.000368	-0.014973	0.011605	0.007802	1.000000	0.009573	0.011500	-0.016643	-0.007480	0.001177	...
...	...	...	...	...	...	...	...	...	...	...	...
date_posted_2019-04-15	-0.013388	-0.010882	-0.007808	0.005422	-0.010626	-0.002675	-0.013906	0.000161	-0.002789	-0.002526	...
date_posted_2019-04-16	-0.001943	0.004671	-0.004073	0.000464	-0.011628	-0.000101	-0.009247	-0.014916	-0.002395	-0.002169	...
date_posted_2019-04-17	-0.001297	0.007489	0.009274	-0.005261	-0.009243	0.004826	-0.005059	-0.003854	0.022321	-0.002496	...
date_posted_2019-04-18	-0.005004	0.005610	-0.007916	-0.000243	0.008394	0.009989	-0.003053	-0.001803	-0.002951	-0.002673	...
date_posted_2019-04-19	0.001468	0.006708	0.004794	0.002272	-0.012796	-0.009645	0.001311	-0.016948	0.022665	-0.002465	...

392 rows × 392 columns

Figure 4.20: linear regression 25000

	price	model_year	cylinders	odometer	days_listed	condition_fair	condition_good	condition_like new	condition_new	condition_salvage	...
price	1.000000	0.430031	0.293487	-0.418083	0.002341	-0.158333	-0.099249	0.142143	0.089263	-0.035849	...
model_year	0.430031	1.000000	-0.149999	-0.472805	-0.004062	-0.222133	-0.154133	0.131099	0.047446	-0.025576	...
cylinders	0.293487	-0.149999	1.000000	0.088469	0.005306	0.034118	0.053629	-0.028491	0.006018	-0.014595	...
odometer	-0.418083	-0.472805	0.088469	1.000000	0.005526	0.181957	0.181067	-0.179733	-0.063468	0.018428	...
days_listed	0.002341	-0.004062	0.005306	0.005526	1.000000	0.004291	0.001050	-0.005559	-0.008585	-0.003277	...
...	...	...	...	...	...	...	...	...	...	...	...
date_posted_2019-04-15	-0.011014	-0.014946	-0.008156	0.004085	-0.009484	0.000010	-0.003086	0.000246	-0.003082	-0.002648	...
date_posted_2019-04-16	-0.002256	0.005345	-0.003188	-0.002375	-0.014207	-0.004983	-0.003544	-0.013451	-0.002686	-0.002308	...
date_posted_2019-04-17	-0.000074	0.006573	0.009803	-0.000748	-0.001148	0.002261	-0.000158	0.002000	0.009423	-0.002370	...
date_posted_2019-04-18	0.001139	0.002185	-0.000273	0.001675	0.004579	0.003532	-0.005061	0.000437	-0.003066	0.010126	...
date_posted_2019-04-19	-0.000965	0.004800	0.006513	0.005836	-0.007064	-0.005070	-0.001301	-0.004284	0.009721	-0.002324	...

392 rows × 392 columns

Figure 4.21: linear regression 50000

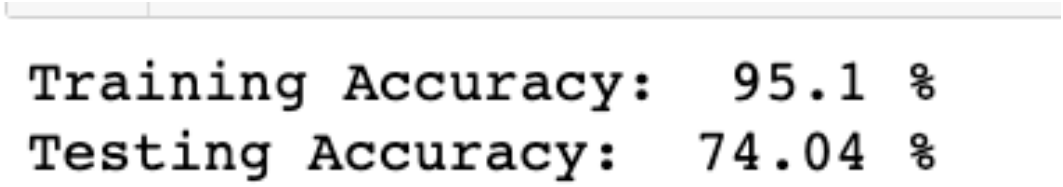

Figure 4.22: linear regression 5000

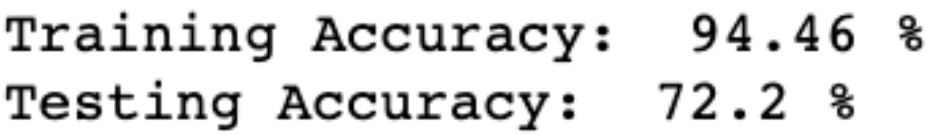

Figure 4.23: linear regression 25000

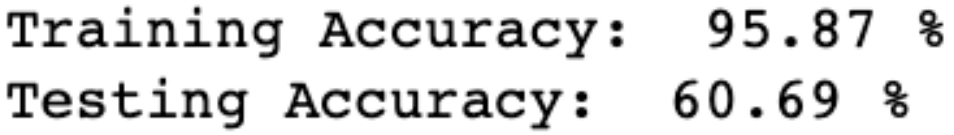

Figure 4.24: linear regression 50000