Machine Learning Final Project

The price prediction of the used cars in Germany since 2015

0416324 胡安鳳
0416308 林正偉
0316323 薛世恩
0416024 陳羿豐

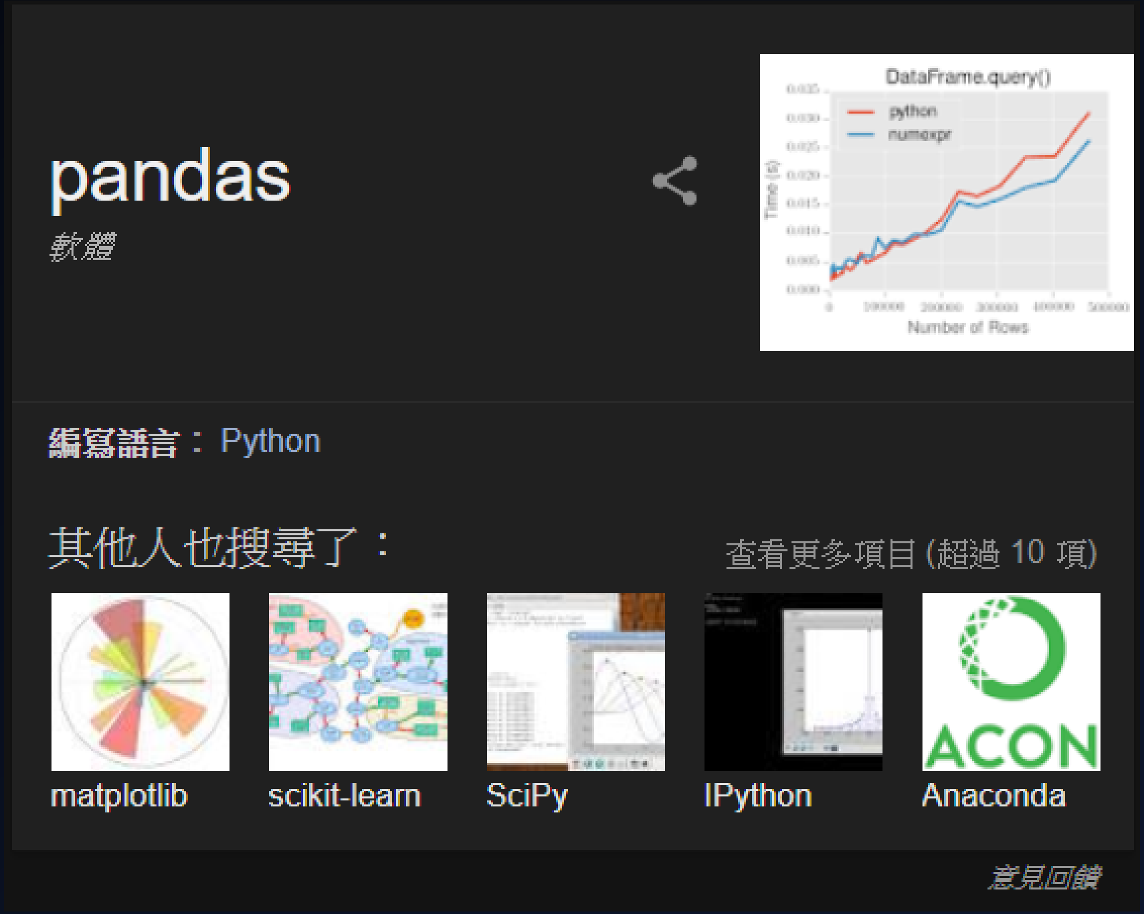
Responsibility assignment

- 胡安鳳 Finding the appropriate dataset, preprocessing(filter the unnecessary data with pandas)

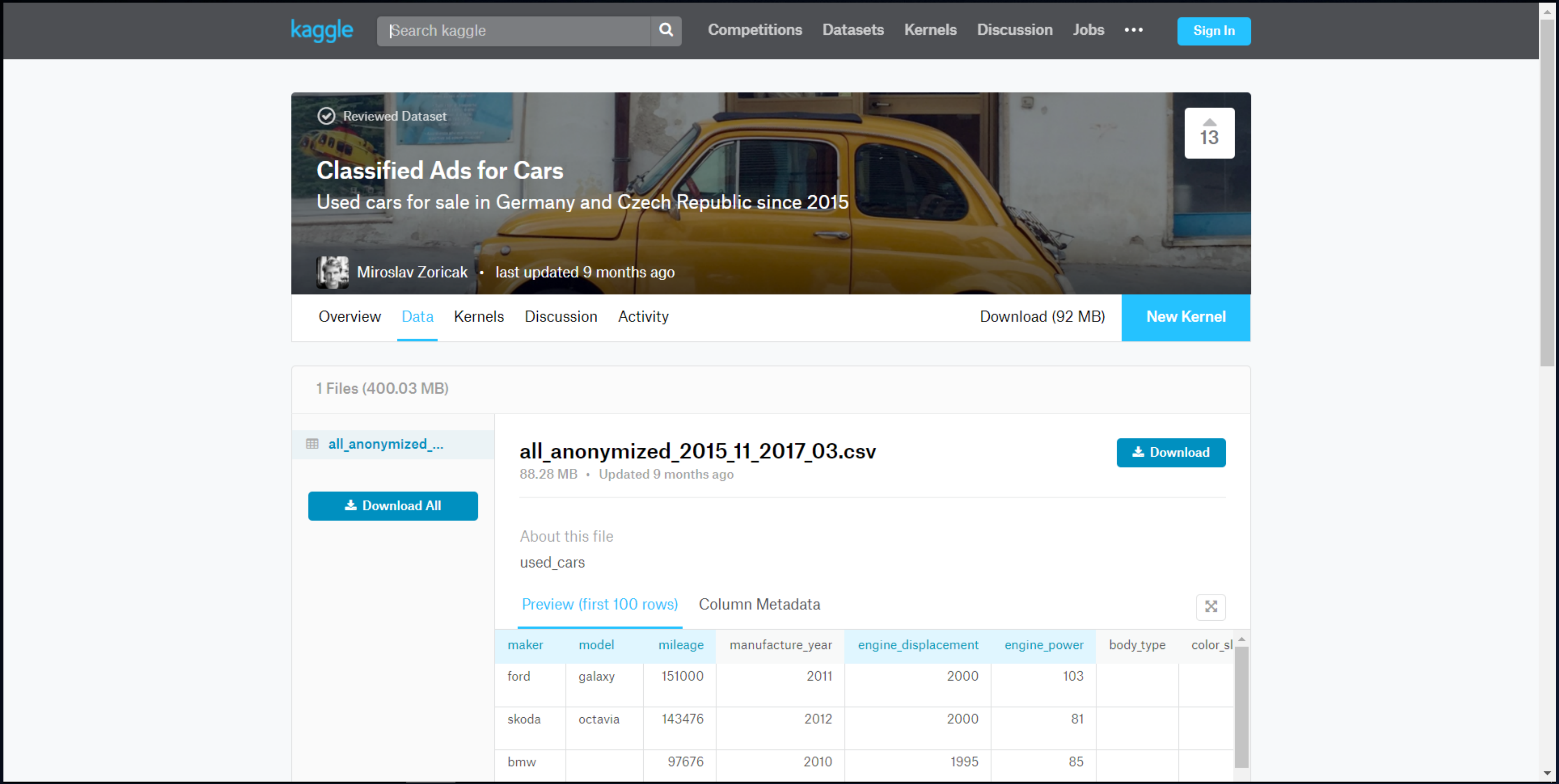
  KNN Regressor analysis, report
- 林正偉 DecisionTreeRegressor analysis, random data generation, cov calculation
- 薛世恩 PDF Plotting, Finding dataset and SVM analysis
- 陳羿豐 Naïve Bayes

Implementation tools

- Pandas for data processing
- Sklearn for ML models
- Matplotlib for graphing
- Python with Anaconda
- Jupyter Notebook for writing the real-time code checking



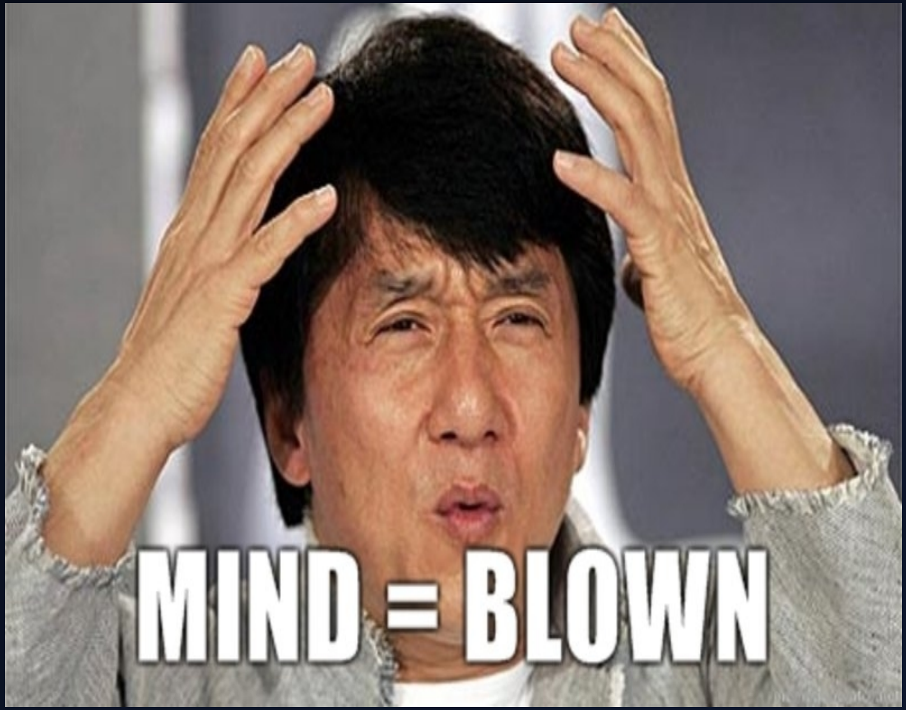
Data Source from Kaggle



SO MUCH DATA!!!

- Data matrix being row=3552912 col=16

```
3552905 skoda,octavia,0,2010,,,other,,,man,,,,2017-03-16 18:57:32.568
3552906 skoda,octavia,,2007,2000,125,other,,,,,,,2017-03-16 18:57:33.
3552907 skoda,octavia,272000,2004,2000,103,other,,,man,,,,2017-03-16 
3552908 skoda,octavia,168000,1999,,,other,,,,,,,2017-03-16 18:57:34.8
3552909 skoda,roomster,54000,2013,1200,63,other,,,,,,,2017-03-16 18:5
3552910 skoda,felicia,,2000,,50,other,,,,,,electric,2017-03-16 18:57:
3552911 skoda,octavia,230000,2006,1900,100,other,,,,,,,2017-03-16 18:
3552912 skoda,fabia,,2001,,,other,,,,,,,2017-03-16 18:57:43.595523+00,
3552913 mercedes-benz,,,,,,other,,,,,,,2017-03-16 19:22:23.946774+00,
```



What attributes to filter and left?

model	mileage	manufacture_year	engine_displacement	engine_power	body_type	color_slug	stk_year	transmission	door_count	seat_count	fuel_type	date_created	date_last_seen	price_eur
galaxy	151000	2011	2000	103			None	man	5	7	diesel	2015-11-14 18:10:06.838319+00	2016-01-27 20:40:15.46361+00	10584.75
octavia	143476	2012	2000	81			None	man	5	5	diesel	2015-11-14 18:10:06.853411+00	2016-01-27 20:40:15.46361+00	8882.31
	97676	2010	1995	85			None	man	5	5	diesel	2015-11-14 18:10:06.861792+00	2016-01-27 20:40:15.46361+00	12065.06
fabia	111970	2004	1200	47			None	man	5	5	gasoline	2015-11-14 18:10:06.872313+00	2016-01-27 20:40:15.46361+00	2960.77
fabia	128886	2004	1200	47			None	man	5	5	gasoline	2015-11-14 18:10:06.880335+00	2016-01-27 20:40:15.46361+00	2738.71
fabia	140932	2003	1200	40			None	man	5	5	gasoline	2015-11-14 18:10:06.894643+00	2016-01-27 20:40:15.46361+00	1628.42
fabia	167220	2001	1400	74			None	man	5	5	gasoline	2015-11-14 18:10:06.915376+00	2016-01-27 20:40:15.46361+00	2072.54
	148500	2009	2000	130			None	auto	5	5	diesel	2015-11-14 18:10:06.924123+00	2016-01-27 20:40:15.46361+00	10547.74
octavia	105389	2003	1900	81			None	man	5	5	diesel	2015-11-14 18:10:06.936239+00	2016-01-27 20:40:15.46361+00	4293.12
	301381	2002	1900	88			None	man	5	5	diesel	2015-11-14 18:10:06.954319+00	2016-01-27 20:40:15.46361+00	1332.35
	202136	2002	1400	55			None	man	5	5	gasoline	2015-11-14 18:10:06.962458+00	2016-01-27 20:40:15.46361+00	740.19
	263840	1998	1900	81			None	man	5	5	diesel	2015-11-14 18:10:06.993167+00	2016-01-27 20:40:15.46361+00	999.26
	105394	2000	1360	55			None	man	3	5	gasoline	2015-11-14 18:10:07.036951+00	2016-01-27 20:40:15.46361+00	1665.43
favorit	41250	1990	1300	44			None	man	5	5	gasoline	2015-11-14 18:10:07.051147+00	2016-01-27 20:40:15.46361+00	370.1
swift	122100	2003	1000	39			None	man	5	5	gasoline	2015-11-14 18:10:07.116629+00	2016-01-27 20:40:15.46361+00	999.26

As we can see from last page

- Body_type and color_slug all contain the empty data
- Most (about99%)of the stk_year are None, which is useless in analysis
- As prediction only for car price but not for the time-serial data, attributes with time can be eliminated but manufactured_year

Remained attribute explanation

Attributes that may affect the price

- Maker/manf’ d year: The manufacturer of that car and such year being manufactured
- Model: Model of the car
- Mileage: Distance the car has been driven
- Engine_displ: swept volume of all the pistons inside the cylin ders of a reciprocating engine
- Engine_power: HP of such engine
- Door and seat count
- Fuel_type: diesel or gasoline?
- Transmission : man or auto

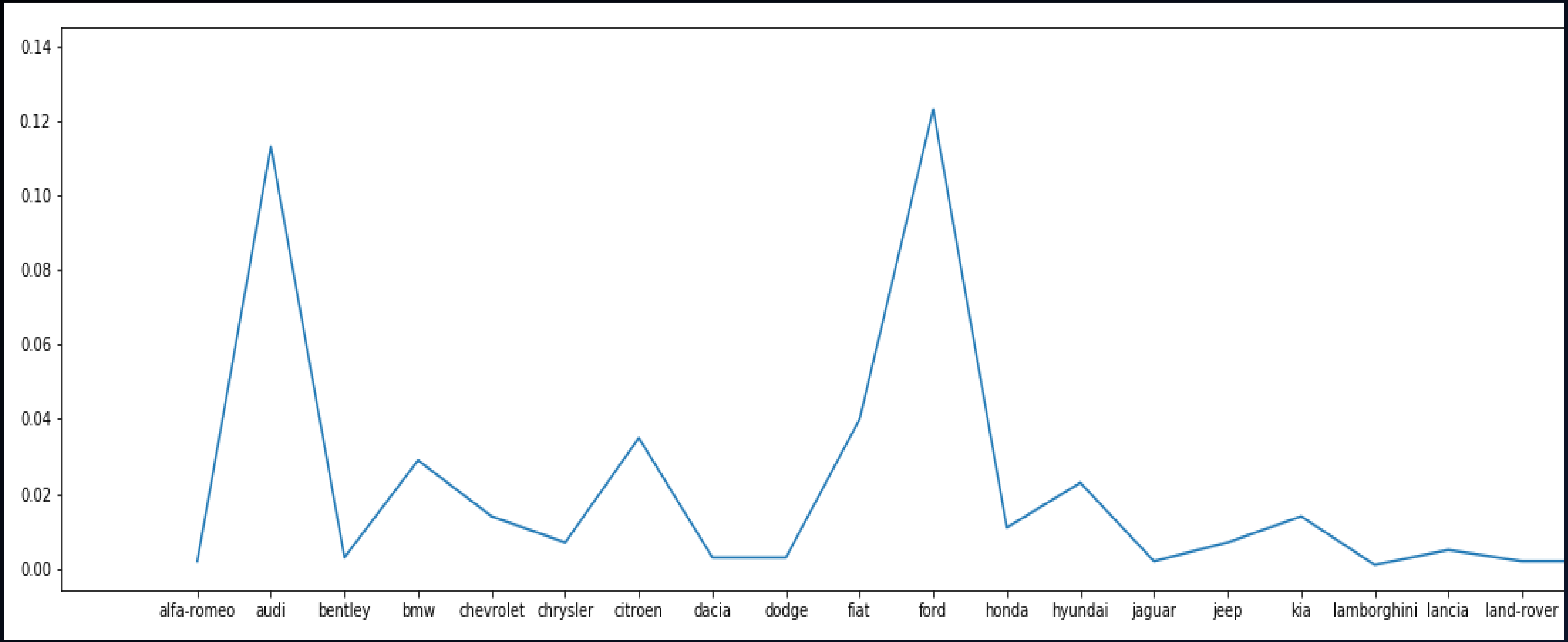
After data preprocessing

- Amount of train = 316162 ,amount of test = 135498 (7:3) col =10

	A	B	C	D	E	F	G	H	I	J	K
316135	opel	meriva	24750	2015	1364	103	man	4	5	gasoline	12563.92
316136	audi	a4	184200	2003	1896	96	man	5	5	diesel	4803.85
316137	audi	a3	225000	2006	1896	77	man	2	5	diesel	5990.67
316138	opel	corsa	115800	1994	1195	33	man	2	5	gasoline	1991.19
316139	opel	adam	9997	2014	1398	74	man	2	4	gasoline	9160.33
316140	skoda	octavia	147800	2008	1896	77	man	5	5	diesel	1295.34
316141	ford	mondeo	253771	2005	1997	85	auto	4	5	diesel	2200.81
316142	audi	a6	95700	2013	2967	230	auto	4	5	diesel	37822.39
316143	skoda	octavia	187364	2005	1900	66	man	5	5	diesel	3330.87
316144	honda	accord	143900	2000	1850	100	auto	4	5	gasoline	1999.59
316145	kia	sportage	118500	2008	1991	103	man	5	5	diesel	6658.03
316146	fiat	freemont	95000	2012	1956	103	man	4	7	diesel	15507.59
316147	skoda	fabia	7181	2014	1197	63	man	4	5	gasoline	9889.45
316148	audi	a6	215000	2005	2698	132	man	4	5	diesel	7850.3
316149	opel	tigra	128000	1995	1389	66	man	3	4	gasoline	999.22
316150	opel	astra	245000	1999	1796	85	man	4	5	gasoline	1499.15
316151	skoda	octavia	242000	2002	1896	66	man	5	5	diesel	3145.82
316152	toyota	yaris	138134	1999	998	50	man	4	5	gasoline	1750.52
316153	mini	cooper	59362	2012	1598	155	man	2	4	gasoline	20161.51
316154	opel	insignia	53900	2013	1956	118	auto	4	5	diesel	14206.22
316155	skoda	octavia	186400	2007	1896	77	man	5	5	diesel	1295.34
316156	ford	mondeo	142165	2008	2000	103	man	5	5	diesel	7031.83
316157	opel	zafira	122600	2006	1910	110	man	4	7	diesel	9000
316158	ford	focus	259453	2005	1600	66	man	5	5	diesel	2627.68
316159	ford	mondeo	114648	2008	2000	103	man	5	5	diesel	7772.02
316160	jeep	compass	19979	2013	2143	120	man	4	5	diesel	20986.12
316161	dacia	sandero	120000	2008	1390	55	man	5	5	gasoline	1295.34
316162	honda	cr-v	98102	2011	2199	110	auto	4	5	diesel	17495.19

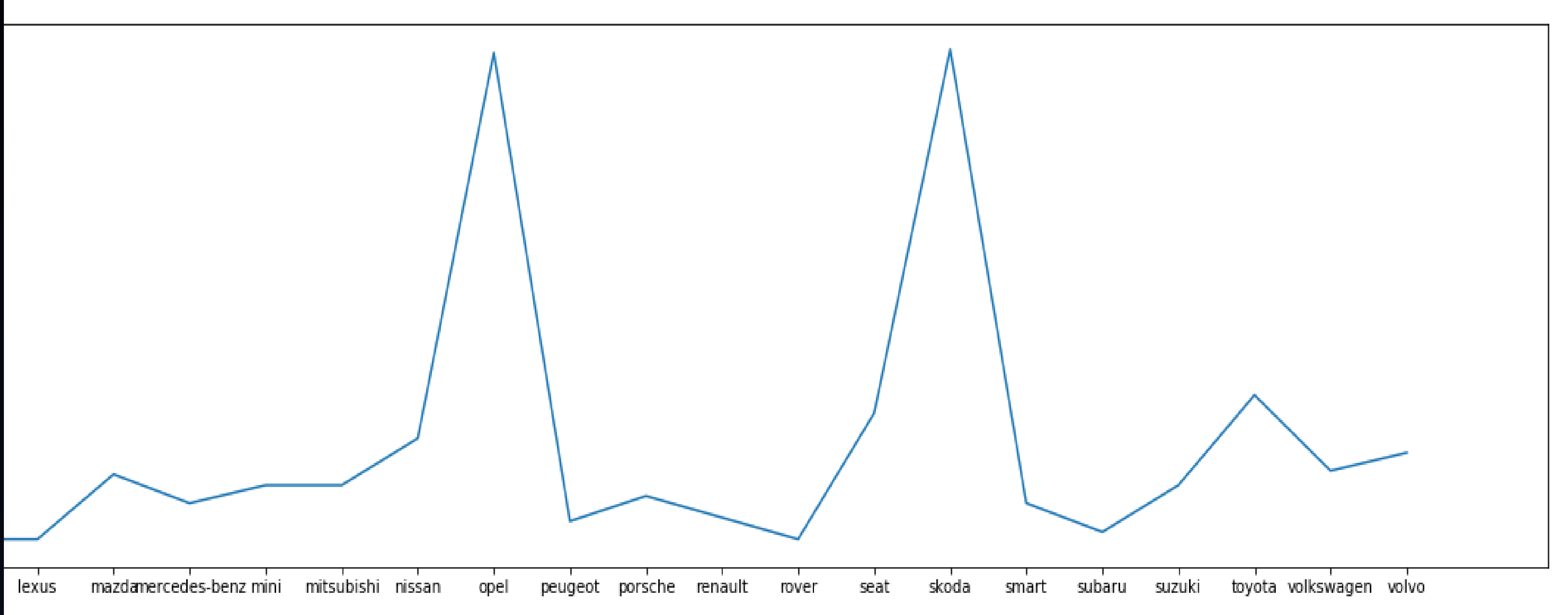
	A	B	C	D	E	F	G	H	I	J	K
135471	opel	astra	187761	2009	1686	81	man	4	5	diesel	4790.71
135472	peugeot	206	205000	1999	1360	55	man	3	4	gasoline	1295.34
135473	renault	twingo	70902	2012	1100	55	man	3	4	gasoline	1295.34
135474	mercedes-	vito	82950	2014	2143	120	auto	4	9	diesel	20885.09
135475	opel	astra	230000	2000	1598	74	man	2	5	gasoline	790.04
135476	kia	rio	15	2014	1396	80	man	4	5	gasoline	13855.14
135477	kia	venga	13000	2012	1396	66	man	4	5	gasoline	10994.89
135478	kia	carens	2500	2014	1591	99	man	4	5	gasoline	18492.04
135479	mazda	6	9000	2014	1998	121	man	4	5	gasoline	19036.82
135480	audi	q5	9000	2015	1968	110	man	4	5	diesel	37420.76
135481	kia	sportage	35100	2012	1995	135	man	4	5	diesel	23001.07
135482	opel	corsa	25000	2010	1398	64	man	5	5	gasoline	7250
135483	opel	mokka	33433	2014	1364	103	man	4	5	gasoline	14668.36
135484	skoda	fabia	144669	2007	1198	51	man	5	5	lpg	1295.34
135485	peugeot	2008	0	2017	1199	81	auto	5	5	gasoline	1295.34
135486	lancia	y	22500	2015	1242	51	man	4	5	gasoline	8771.28
135487	bmw	x5	84200	2011	2993	225	auto	4	7	diesel	32676.09
135488	volvo	740	399999	1987	2300	83	man	4	5	gasoline	1250.74
135489	porsche	911	15	2015	3800	294	auto	2	4	gasoline	112445.6
135490	mini	one	75947	2011	1598	66	man	2	4	diesel	7500.3
135491	porsche	cayenne	50	2015	2967	193	auto	4	5	diesel	92563.29
135492	opel	astra	148000	2005	1598	77	man	4	5	gasoline	3001.67
135493	kia	sportage	15536	2013	1591	99	man	4	5	gasoline	16641.23
135494	seat	ibiza	119000	1999	1390	44	man	3	5	gasoline	1150
135495	peugeot	308	72000	2010	1560	80	man	5	5	diesel	1295.34
135496	fiat	500	5	2015	1242	51	man	2	4	gasoline	10064.51
135497	ford	fusion	153400	2004	1388	59	man	5	5	gasoline	1295.34
135498	fiat	punto	148000	2007	1248	66	man	4	4	diesel	3280.2

PDF of maker

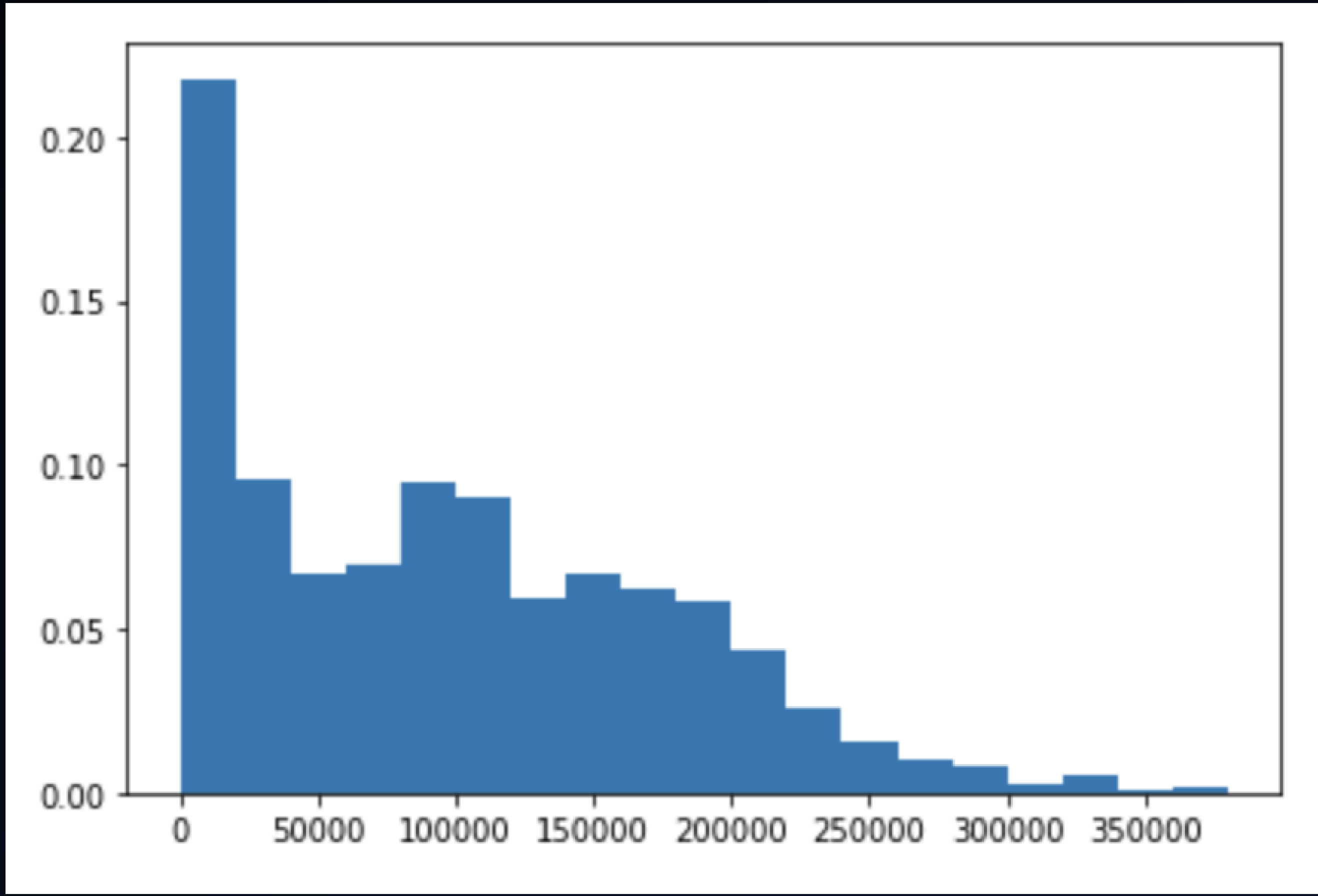


0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
alfa-romeo
audi
bentley
bmw
chevrolet
chrysler
citroen
dacia
dodge
fiat
ford
honda
hyundai
jaguar
jeep
kia
lamborghini
lancia
land-rover

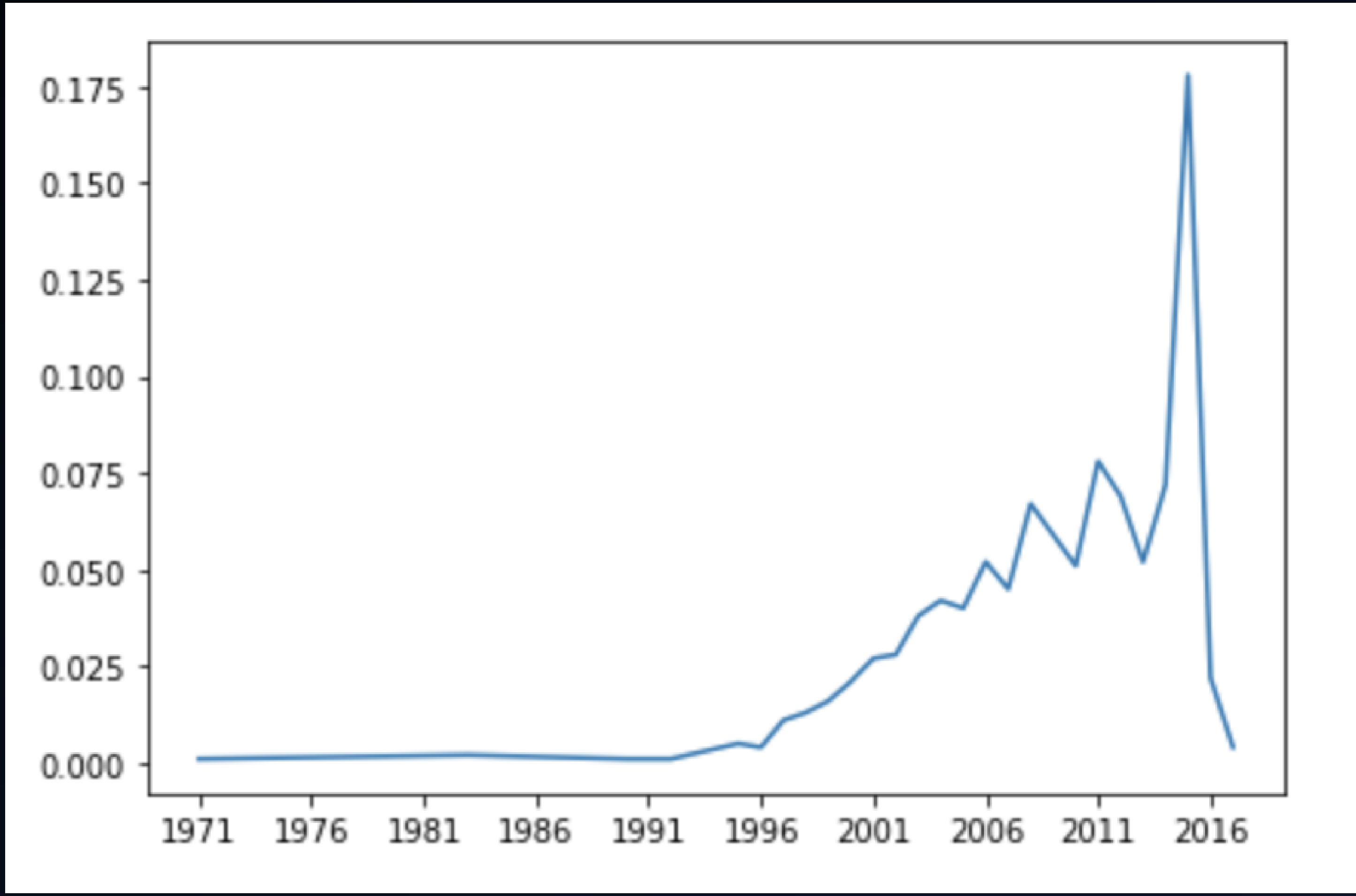
PDF of maker


lexus
mazda
mercedes-benz
mini
mitsubishi
nissan
opel
peugeot
porsche
renault
rover
seat
skoda
smart
subaru
suzuki
toyota
volkswagen
volvo

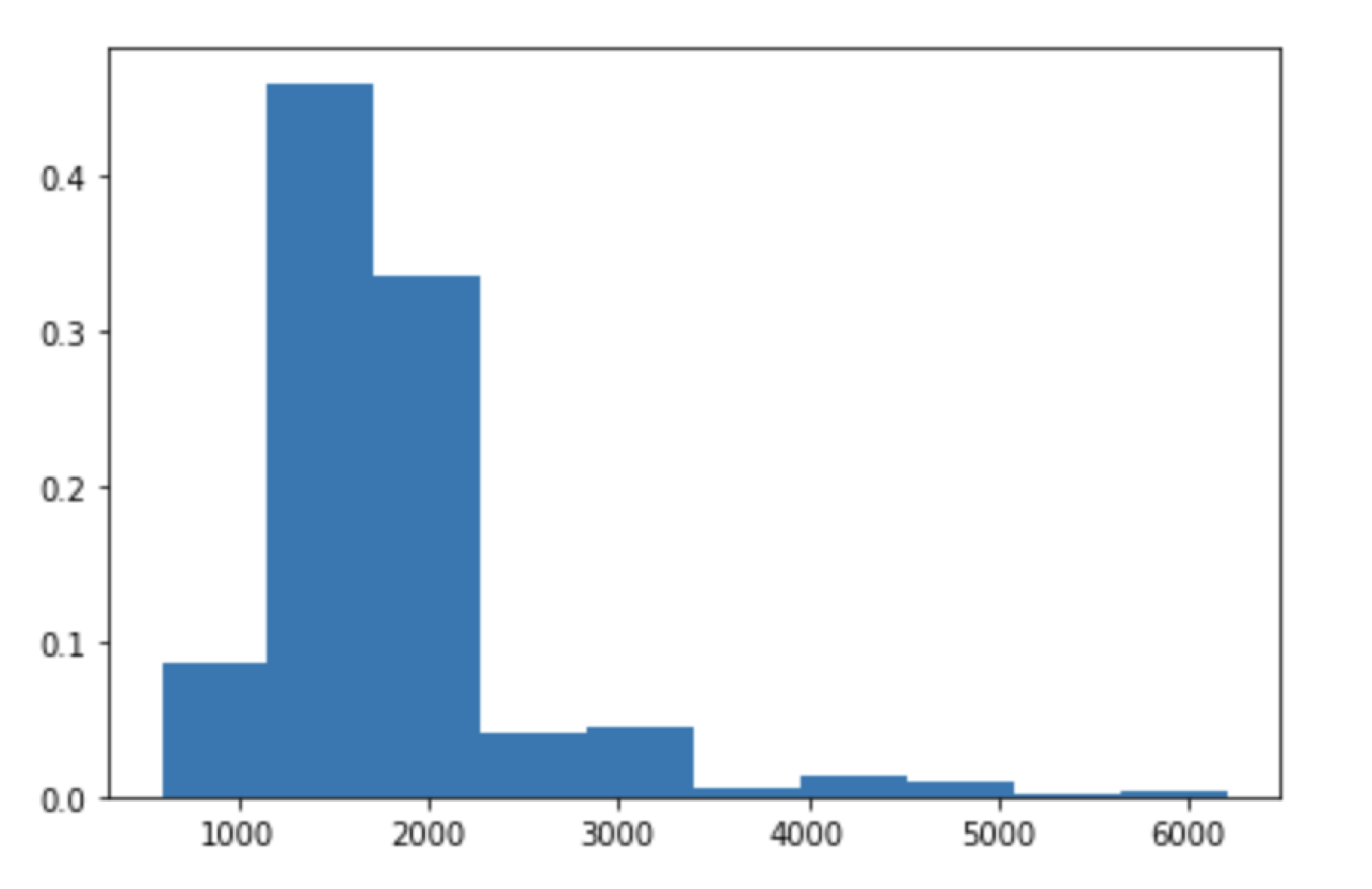
PDF of mileage



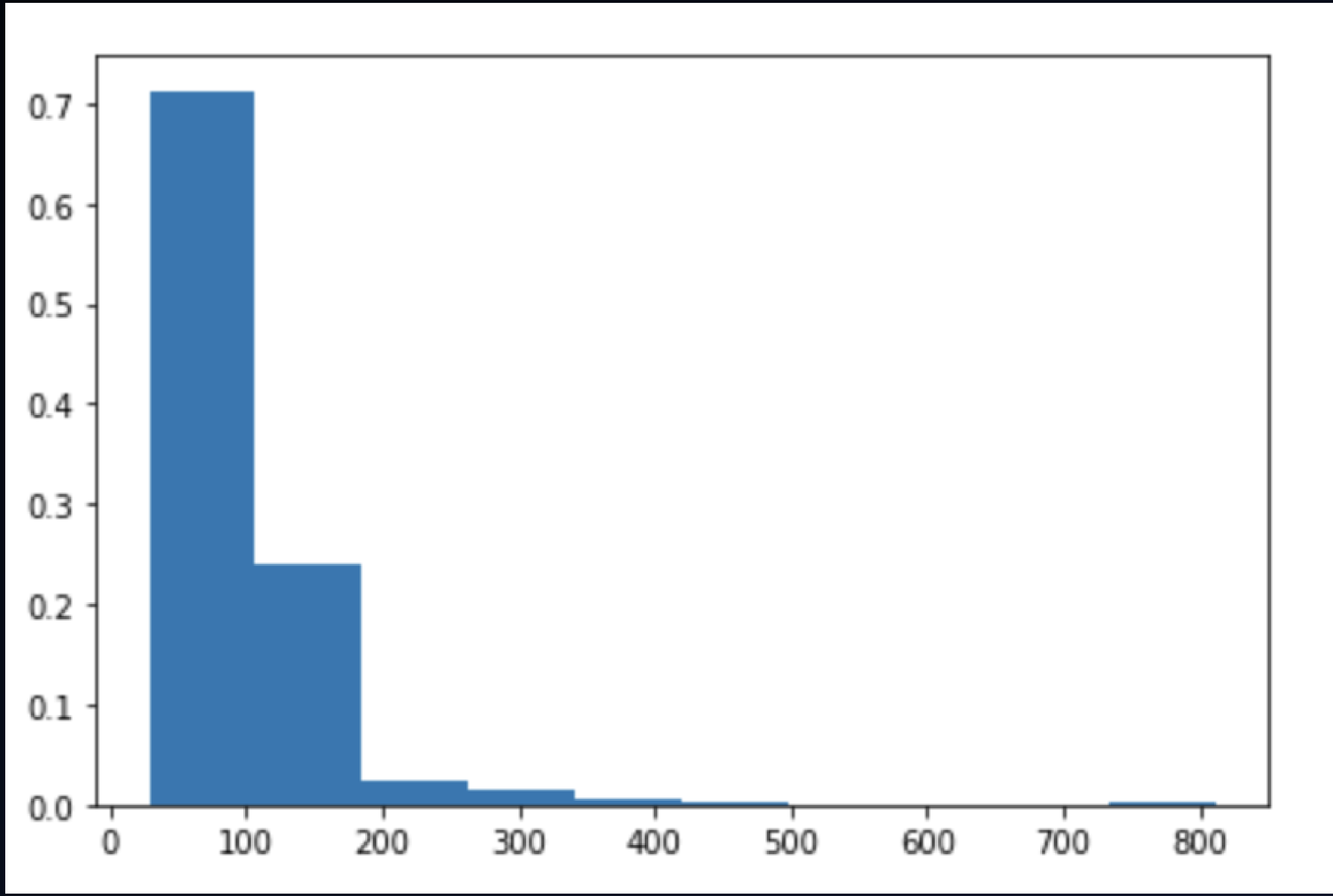
PDF of manufacture year



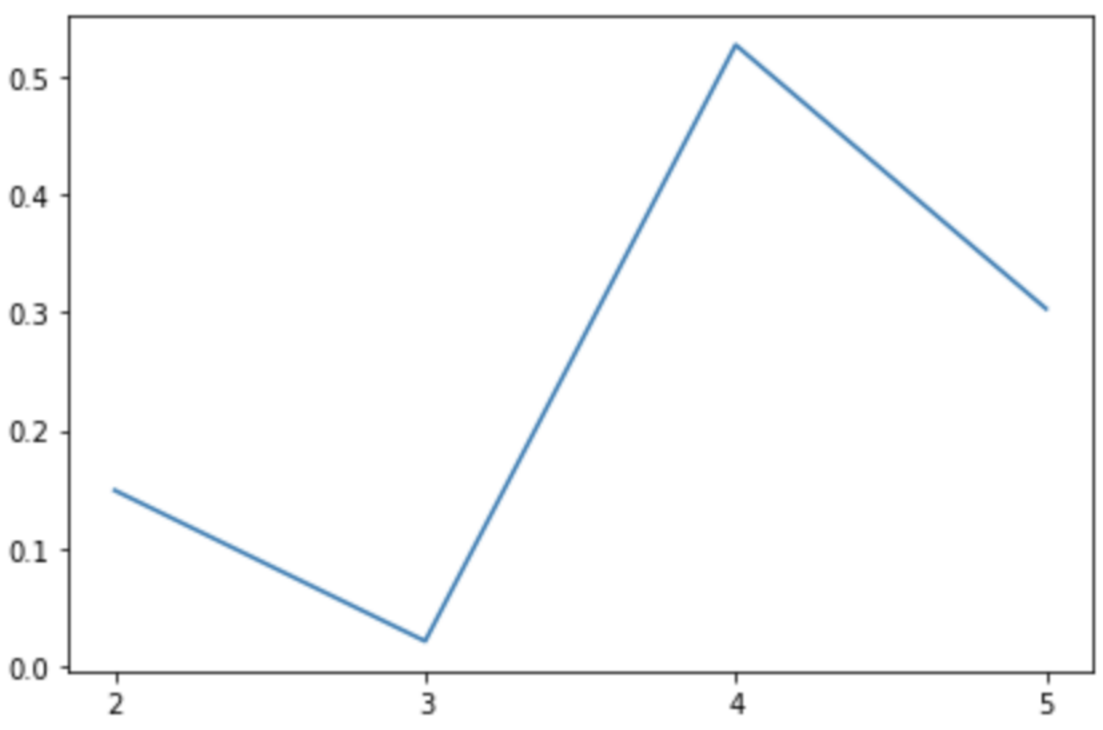
PDF of engine displacement

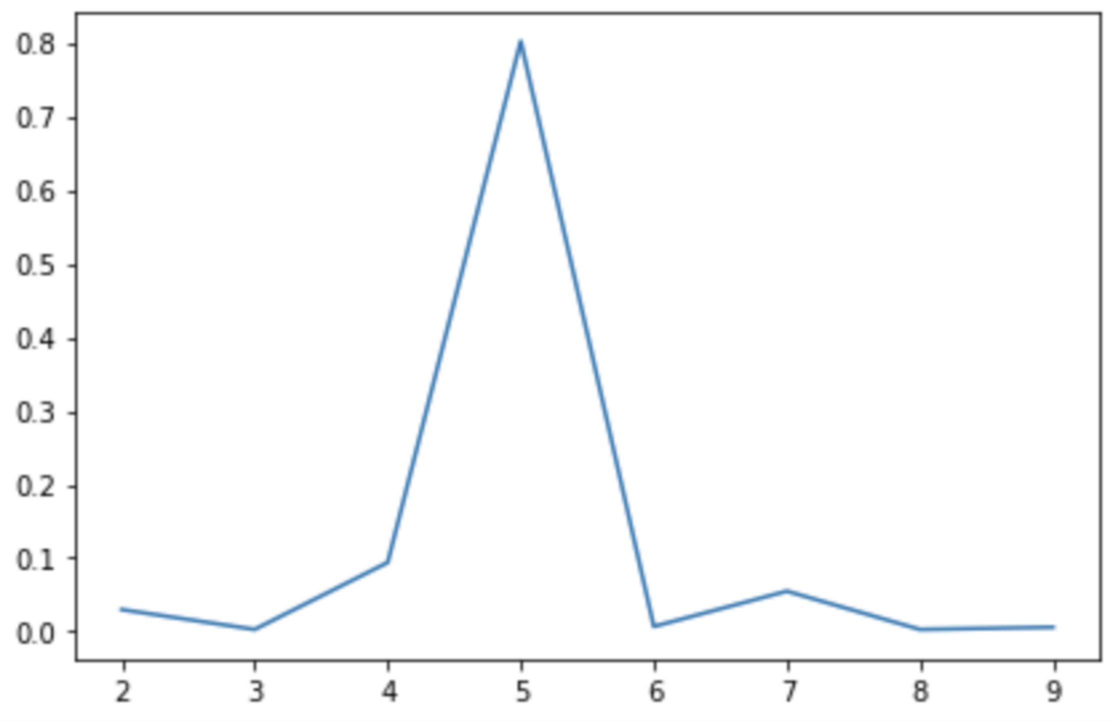


PDF of engine power

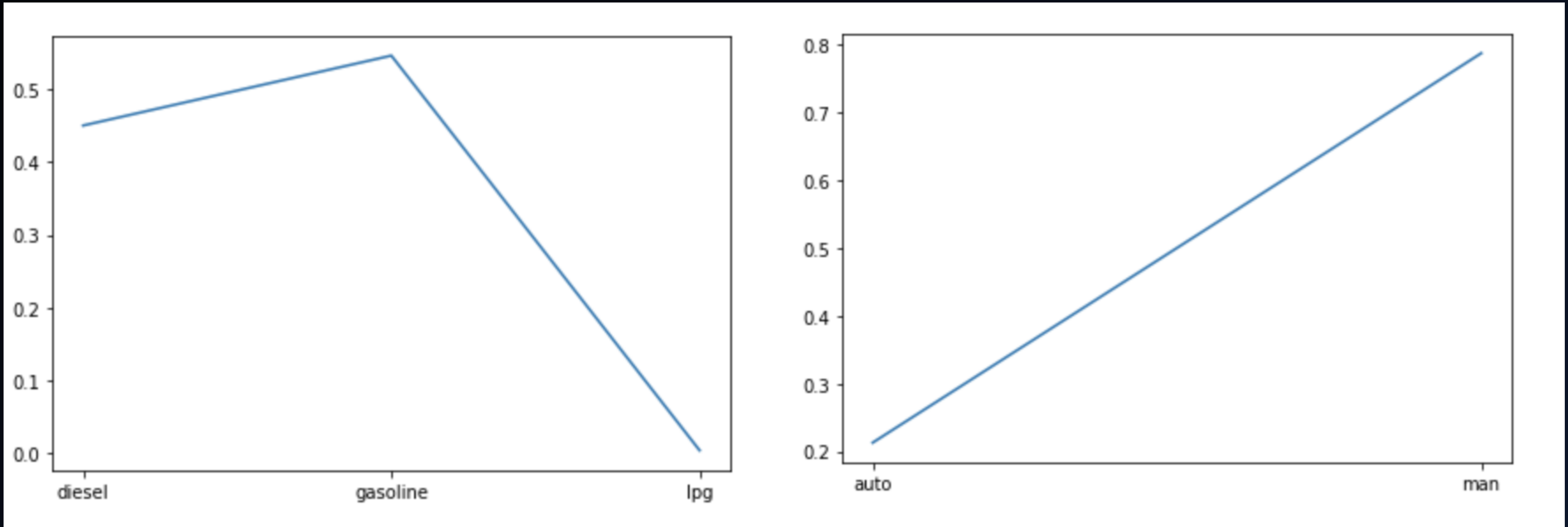


PDF of Door and Seat





PDF of Fuel and transmisson



More detailed ,analyze COV to see what does we REALLY NEED

- Attribute-covariance analysis credit to 0416308 David Lin

```
Correlation coefficient maker price: 0.139980036187
Correlation coefficient model price: 0.160782252636
Correlation coefficient mileage price: -0.103459401042
Correlation coefficient manufacture_year price: 0.10861649809
Correlation coefficient engine_displacement price: 0.13050695l953
Correlation coefficient engine_power price: 0.191493812909
Correlation coefficient transmission price: 0.110734227529
Correlation coefficient door_count price: -0.0551736306567
Correlation coefficient seat_count price: -0.0307518213302
Correlation coefficient fuel_type price: 0.0326543028783
```

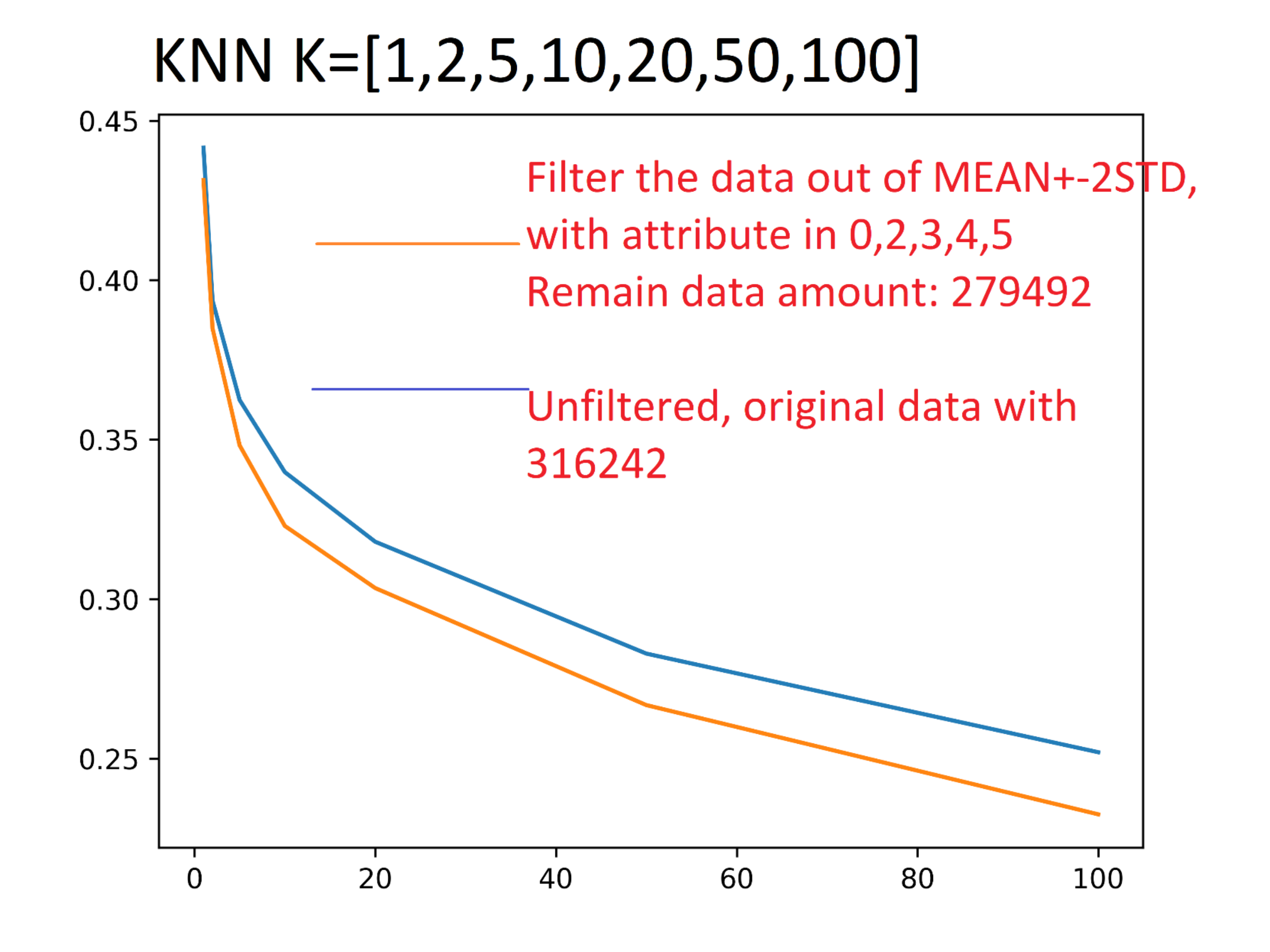
Regressor vs Classifier?

- Continuous output – Regressor
- Discrete output - Classifier

Model1 KNN Regressor by 胡安鳳

```
KNN with K=  1  There is  0.42650076015882155  that the predict price is within  1000  eur of actual price
KNN with K=  2  There is  0.37768823156061343  that the predict price is within  1000  eur of actual price
KNN with K=  5  There is  0.3447504760217863  that the predict price is within  1000  eur of actual price
KNN with K=  10  There is  0.3195545321702165  that the predict price is within  1000  eur of actual price
KNN with K=  20  There is  0.29528849134304563  that the predict price is within  1000  eur of actual price
KNN with K=  50  There is  0.25657205272402545  that the predict price is within  1000  eur of actual price
KNN with K=  100  There is  0.22482988678799687  that the predict price is within  1000  eur of actual price
[1, 2, 5, 10, 20, 50, 100] [0.42650076015882155, 0.37768823156061343, 0.3447504760217863, 0.3195545321702165, 0.29528849134304563, 0.2565720527240
2545, 0.22482988678799687]
 alfons@alfons  ~/Desktop/Programming/Machine_Learning_Fall_2017/Final_proj  master ●  python KNN_regressor.py
Training_data count  272347
KNN with K=  1  There is  0.42698047203648765  that the predict price is within  1000  eur of actual price
KNN with K=  2  There is  0.37860337421954565  that the predict price is within  1000  eur of actual price
KNN with K=  5  There is  0.34016738254439177  that the predict price is within  1000  eur of actual price
KNN with K=  10  There is  0.31531092709855496  that the predict price is within  1000  eur of actual price
KNN with K=  20  There is  0.2948530605617795  that the predict price is within  1000  eur of actual price
KNN with K=  50  There is  0.25756837739302424  that the predict price is within  1000  eur of actual price
KNN with K=  100  There is  0.22373023956073151  that the predict price is within  1000  eur of actual price
[1, 2, 5, 10, 20, 50, 100] [0.42698047203648765, 0.37860337421954565, 0.34016738254439177, 0.31531092709855496, 0.2948530605617795, 0.257568377393
02424, 0.22373023956073151]
 alfons@alfons  ~/Desktop/Programming/Machine_Learning_Fall_2017/Final_proj  master ●  python KNN_regressor.py
Training_data count  285624
KNN with K=  1  There is  0.42810225981195293  that the predict price is within  1000  eur of actual price
KNN with K=  2  There is  0.38236726741354116  that the predict price is within  1000  eur of actual price
KNN with K=  5  There is  0.3495475947984472  that the predict price is within  1000  eur of actual price
KNN with K=  10  There is  0.3259236298690756  that the predict price is within  1000  eur of actual price
KNN with K=  20  There is  0.30300816248210305  that the predict price is within  1000  eur of actual price
KNN with K=  50  There is  0.2648230970198822  that the predict price is within  1000  eur of actual price
KNN with K=  100  There is  0.2332801960176534  that the predict price is within  1000  eur of actual price
[1, 2, 5, 10, 20, 50, 100] [0.42810225981195293, 0.38236726741354116, 0.3495475947984472, 0.3259236298690756, 0.30300816248210305, 0.2648230970198
822, 0.2332801960176534]
 alfons@alfons  ~/Desktop/Programming/Machine_Learning_Fall_2017/Final_proj  master ●  python KNN_regressor.py
Training_data count  279492
KNN with K=  1  There is  0.43109123381894937  that the predict price is within  1000  eur of actual price
KNN with K=  2  There is  0.3846182231472051  that the predict price is within  1000  eur of actual price
KNN with K=  5  There is  0.34835200519564863  that the predict price is within  1000  eur of actual price
KNN with K=  10  There is  0.3229420360448125  that the predict price is within  1000  eur of actual price
KNN with K=  20  There is  0.3032590886950361  that the predict price is within  1000  eur of actual price
KNN with K=  50  There is  0.26682312654061313  that the predict price is within  1000  eur of actual price
KNN with K=  100  There is  0.23250527683065433  that the predict price is within  1000  eur of actual price
[1, 2, 5, 10, 20, 50, 100] [0.43109123381894937, 0.3846182231472051, 0.34835200519564863, 0.3229420360448125, 0.3032590886950361, 0.26682312654061
313, 0.23250527683065433]
```

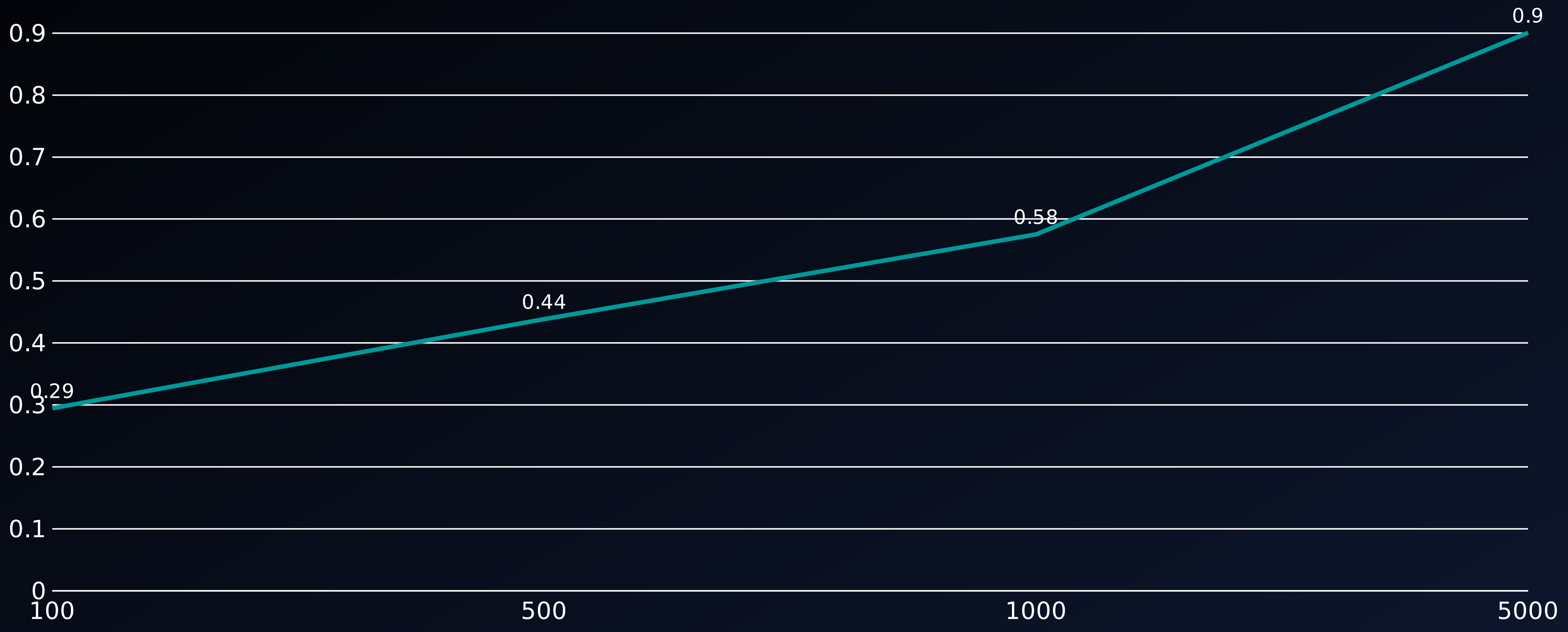
Model1 KNN Regressor by 胡安鳳



The contribution of one training sample-accuracy from 0.441/316242 = 1.39e-6
To
0.431/279492 = 1.542e-6

Enhance the accuracy of unit training sample

Model 2 DT Regressor by 林正偉



```
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 DT_no_sort.py
There is 0.28793044915792115 that the predict price is within 100 eur of actual price
There is 0.4318956737368817 that the predict price is within 500 eur of actual price
There is 0.5693884780587167 that the predict price is within 1000 eur of actual price
There is 0.8942788823451269 that the predict price is within 5000 eur of actual price
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 data_generator.py
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 DT_no_sort.py
There is 0.29012236342971853 that the predict price is within 100 eur of actual price
There is 0.43227944323901457 that the predict price is within 500 eur of actual price
There is 0.5710121182600482 that the predict price is within 1000 eur of actual price
There is 0.8947954951364596 that the predict price is within 5000 eur of actual price
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 data_generator.py
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 DT_no_sort.py
There is 0.2891850802225863 that the predict price is within 100 eur of actual price
There is 0.431947335016015 that the predict price is within 500 eur of actual price
There is 0.570052694504716 that the predict price is within 1000 eur of actual price
There is 0.894153419238603 that the predict price is within 5000 eur of actual price
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 data_generator.py
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 DT_no_sort.py
There is 0.286948884854389 that the predict price is within 100 eur of actual price
There is 0.43115027528081595 that the predict price is within 500 eur of actual price
There is 0.5707316713161817 that the predict price is within 1000 eur of actual price
There is 0.8950464213493926 that the predict price is within 5000 eur of actual price
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 data_generator.py
davidlin@davidlin-X555LJ:/media/davidlin/Data/Course/Junior_1/Machine_Learning/Project$ python3.5 DT_no_sort.py
There is 0.2883732601219206 that the predict price is within 100 eur of actual price
There is 0.4305008192002834 that the predict price is within 500 eur of actual price
There is 0.5680010037048517 that the predict price is within 1000 eur of actual price
There is 0.8955261332270587 that the predict price is within 5000 eur of actual price
```

Model 3 Random Forest Regressor by 薛世恩

```
error < 1000:  0.4549621024819738
error < 1500:  0.5710901348369336
error < 2000:  0.6543244499878226
```

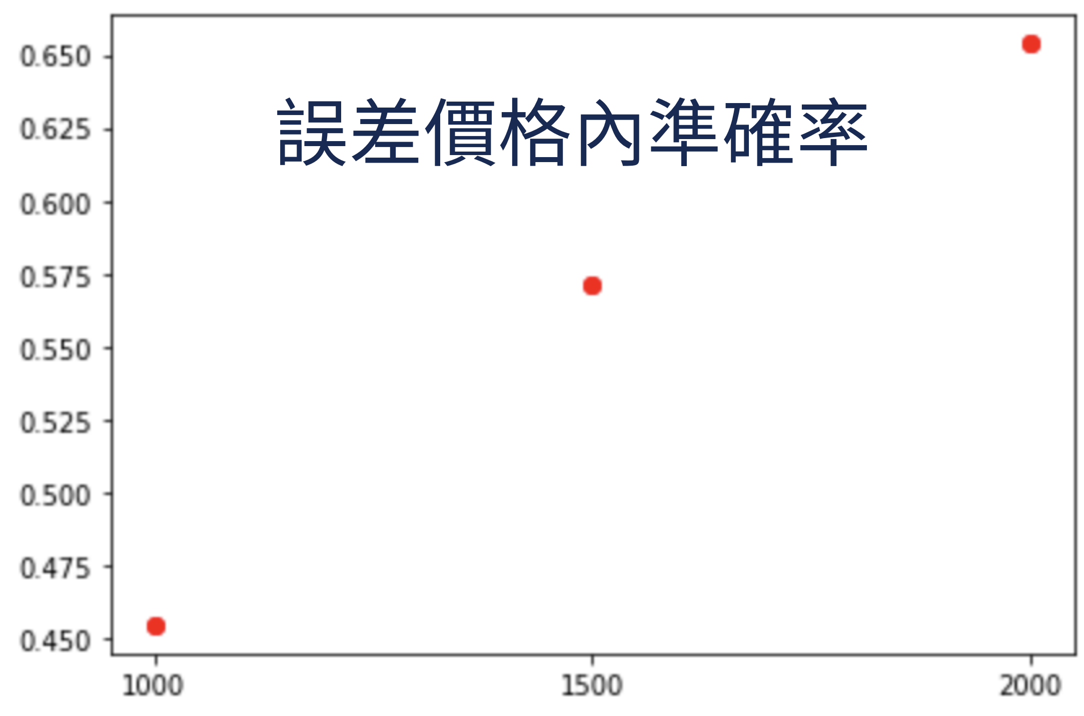
```
max_depth = 5, error < 2000:  0.4347623932633195
max_depth = 6, error < 2000:  0.4954648424009385
max_depth = 7, error < 2000:  0.5341741883584138
```

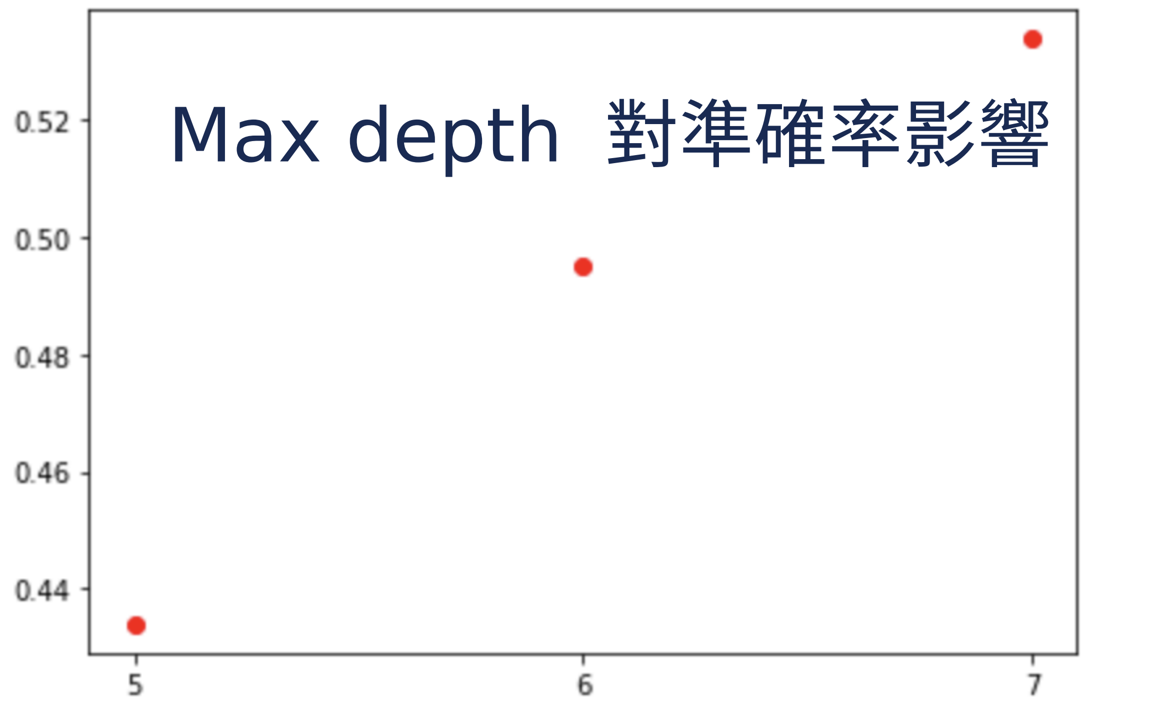
```
256.872
-2174.025
-30.502
3438.085
1468.6731452
684.995
265.786
```

max_depth : integer or None, optional (default=None)

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Model 3 Random Forest Regressor





Model 4 Naïve Bayes Classifier by 陳羿豐

- 因為 Naive Bayes 不能產生連續型輸出 (不能作為 regressor) ，所以我就把價格按照位數做分類。
- 我發現價格最便宜的還不到 1 歐元，最貴的超過一百萬歐元。所以我就分類成：不到 1 、 1 ～ 10 、 10 ～ 100 、……、十萬～一百萬、超過一百萬，共 8 類
- 因為上一份作業，我用 scikit-learn 套件做 Naive Bayes ，結果非常的糟。我覺得我被套件雷了……
- 於是我下定決心，要寫一個自己的 Naive Bayes ，而且要用最強大的—— Java ！

Functionalities

- 我的程式可處理連續型和離散型特徵。使用到的特徵：
- 離散型：製造商、車的型號、門的數量、座椅數量、燃料種類、變速器種類（自排 / 手排）
- 連續型：里程數、製造年份、引擎 cc 數、引擎馬力
- 我假設連續型特徵符合常態分佈

Thank you for listening

- Source code can be found at:

https://github.com/Alfons0329/Machine_Learning_Fall_2017/tree/master/Final_proj

- README.md will be added later in winter vacation.
- Great thanks to all the teammate who contribute passionately to this final project.