Dataset name: Utrecht housing dataset

Dataset link: https://www.kaggle.com/datasets/ictinstitute/utrecht-housing-dataset?select=utrechthousinghuge.csv

Import the dataset

```
df = pd.read_csv('gdrive/My Drive/Colab Notebooks/utrechthousinghuge.csv')
```

Data cleaning

First of all , we will see the shape of the dataset , which is :

```
(2499, 16)
```

This dataset has 2499 rows and 16 columns , or features .

Drop duplicates

We drop the duplicates and then we show the shape again , to see if there were duplicates in the dataset.

```
df.drop_duplicates()
```

```
(2499, 16)
```

There were not duplicates , since the rows didn't changed .

Drop NA variables

```
df.dropna(inplace = True)
```

Returns none , since there are not any NaN variables .

Searching for missing data

There is no null data .

```
Isnull:
id              0
zipcode         0
lot-len         0
lot-width       0
lot-area        0
house-area      0
garden-size     0
balcony         0
x-coor          0
y-coor          0
buildyear       0
bathrooms       0
taxvalue        0
retailvalue     0
energy-eff      0
monument        0
dtype: int64
```

Viewing not unique data

There exists not unique data .

```
Nunique:
id              2499
zipcode            4
lot-len          736
lot-width        101
lot-area         381
house-area       170
garden-size      907
balcony            3
x-coor           383
y-coor           927
buildyear         41
bathrooms          2
taxvalue        1380
retailvalue      325
energy-eff         2
monument           2
dtype: int64
```

Choosing the target variable

Until now , we cleaned the data , from now on we will work with machine learning models , we will choose attributes , change location of attributes , dropping features and much more machine learning data.

Putting the attribute in the first column

We will have prediction on the price feature of the house , the retailvalue attribute , based on predictors . We will put it in the first column position to have a greater understanding perspective of the soon to be shown , the correlation matrix , which will help us to choose the predictors in order to use machine learning models .

```
columns = list(df.columns)

columns.remove('retailvalue')
columns.insert(0,'retailvalue')
df = df[columns]
```

Dropping id column

We don't need the id column , we will drop it .

```
df = df.drop(columns=['id'])
```

This is the view of the dataset after we dropped id column.

	retailvalue	zipcode	lot-len	lot-width	lot-area	house-area	garden-size	balcony	x-coor	y-coor	buildyear	bathrooms	taxvalue	energy-eff	monument
0	232000	3520	35.1	14.2	499	147	406	1	2192	5470	1978	2	173100	0	0
1	184000	3520	22.2	18.2	404	109	328	1	2059	5228	1981	1	136000	0	1
2	230000	3520	59.7	11.8	704	89	629	1	2243	5388	1972	1	169000	0	0
3	257000	3520	39.0	18.6	726	130	669	1	2050	5432	1970	2	196900	0	0
4	286000	3520	44.2	18.1	800	160	750	0	2138	5110	1978	2	226000	0	0

Describe the dataset

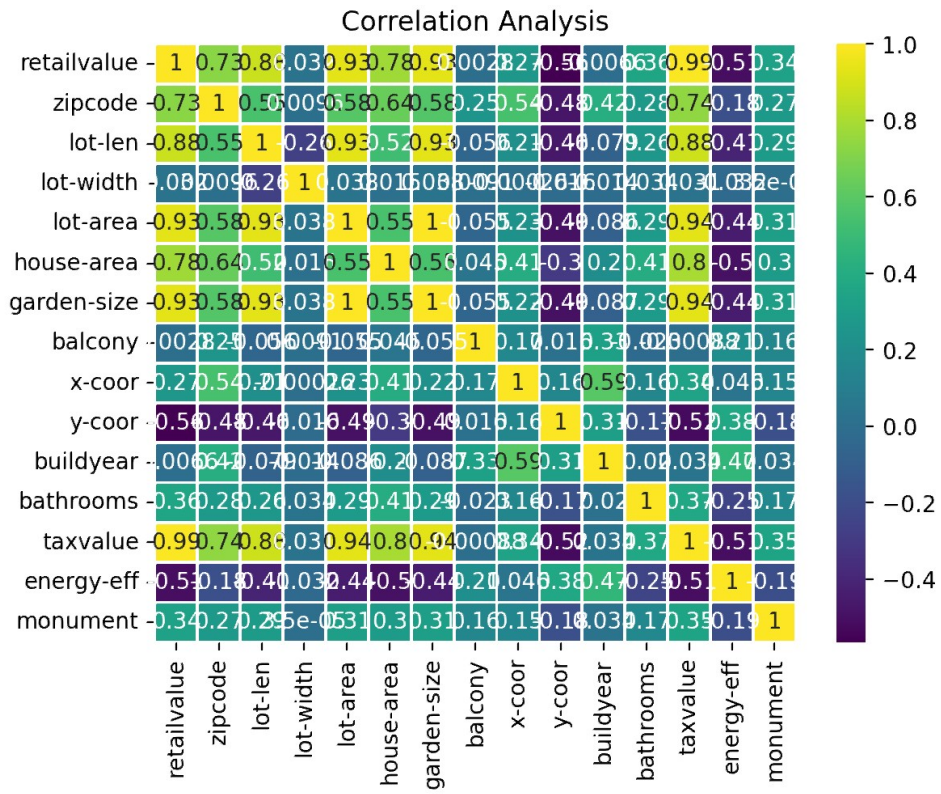
This method , shows us statistical information about the dataset .

	retailvalue	zipcode	lot-len	lot-width	lot-area	house-area	garden-size	balcony	x-coor	y-coor	buildyear	bathrooms	taxvalue	energy-eff	monument
count	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000	2499.000000
mean	219054.821929	3592.691076	34.997919	14.989396	506.726691	138.014406	431.547819	0.737095	2567.934774	5499.337735	1992.401361	1.278912	170083.953581	0.284114	0.20008
std	81695.336086	119.141675	23.414357	2.898105	319.793785	42.984094	320.229223	0.526708	256.478469	280.795227	10.794545	0.448554	74934.022024	0.451081	0.40014
min	91000.000000	3520.000000	5.600000	10.000000	110.000000	80.000000	10.000000	0.000000	2000.000000	5000.000000	1970.000000	1.000000	64500.000000	0.000000	0.00000
25%	137000.000000	3520.000000	11.500000	12.500000	148.000000	103.000000	83.000000	0.000000	2389.000000	5259.000000	1983.000000	1.000000	91300.000000	0.000000	0.00000
50%	212000.000000	3525.000000	33.200000	14.900000	513.000000	128.000000	436.000000	1.000000	2601.000000	5492.000000	1996.000000	1.000000	164800.000000	0.000000	0.00000
75%	282000.000000	3528.000000	51.950000	17.500000	771.500000	164.000000	699.000000	1.000000	2768.500000	5749.000000	2001.000000	2.000000	225800.000000	1.000000	0.00000
max	435000.000000	3800.000000	116.600000	20.000000	1200.000000	250.000000	1144.000000	2.000000	3000.000000	6000.000000	2010.000000	2.000000	357800.000000	1.000000	1.00000

From the data , it should not have outliers , we will plot the data , in order to see that there are outliers or not .

Correlation Matrix

This shows the correlation of the features between them.



Checking attributes correlation for choosing the predictor

We can look at the correlation analysis plot from above and find the correlation data between the features , or we can use code to see what are the best attributes to take as predictors for Univariate and Bivariate type models .

```
retailvalue    1.000000
taxvalue       0.992432
lot-area       0.934471
garden-size    0.933547
lot-len        0.876142
house-area     0.777607
zipcode        0.727055
bathrooms      0.362596
monument       0.335244
x-coor         0.265028
lot-width      0.032310
balcony        0.002792
buildyear     -0.006575
energy-eff    -0.508410
y-coor        -0.564726
Name: retailvalue, dtype: float64
```

We will take lot-area for the first predictor because of 0.93 score , and taxvalue for the second predictor based on the correlation plot because of 0.94 .

According to "https://www.valuepenguin.com/mortgages/what-is-the-assessed-value-of-a-house",

The assessed value of a home is a yearly estimation of your home's worth, determined by your tax district's municipal property assessor. Local tax officials use this value to calculate the property taxes you pay on your home each year.

--

Univariate Linear Regression

We will chose the target variable (y) and predictor (X) , then we will split the data in training data and testing data .

```
y = df.iloc[:,0].values # target

X = df[['lot-area']] # predictor

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=0)
```

Then we will create the model , fit it , then predict the target variable based on testing predictors.

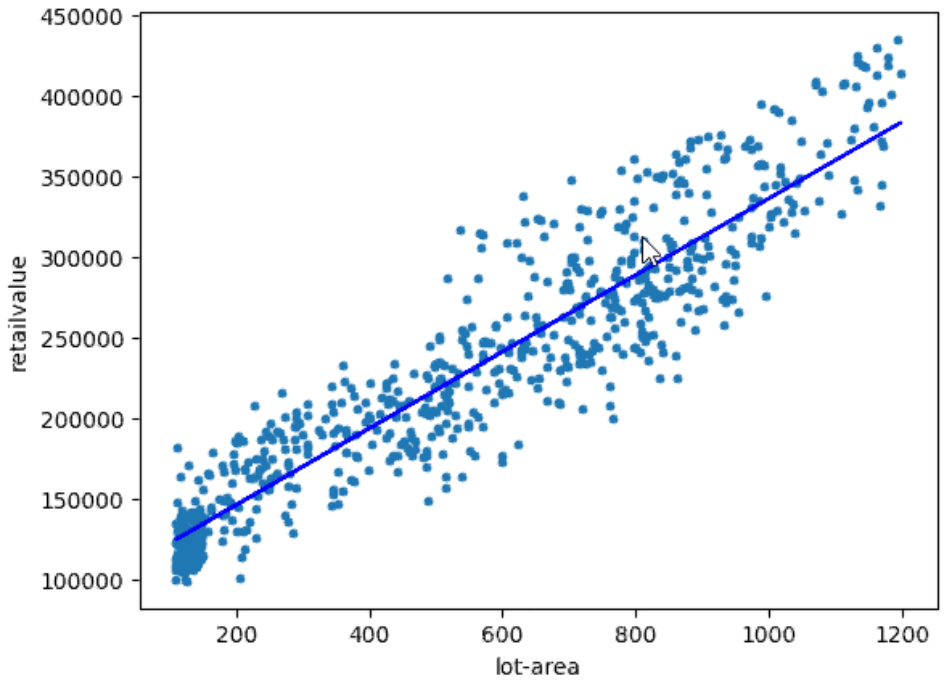
```
regressor = LinearRegression()
regressor.fit(X_train, y_train)

y_pred= regressor.predict(X_test)
```

We will store in a table that is below , the R^2 score , Time , where is the case , cross_validation and mean of the cross_validation scores .

```
intercept_: 98657.36787337426
coef_: [237.51031368]

MAE:  21909.238986809483
MSE:  805103008.1358398
RMSE:  28374.337140025666
R^2 :  0.8839326089985096
Time:  0.015860319137573242
```

Plotting Univariate Linear Regression

There are not any outliers , based on the plot .



--

--

Bivariate Linear Regression

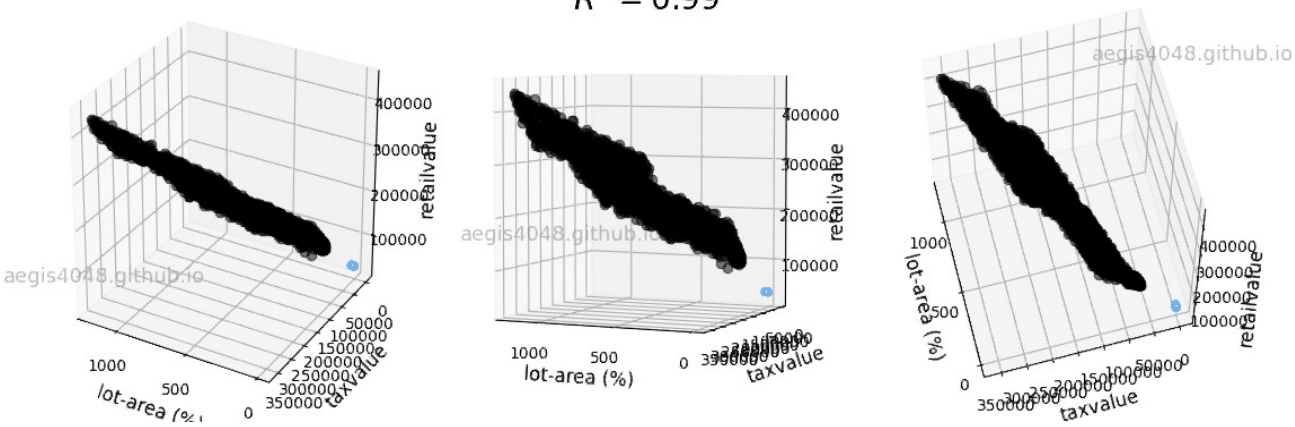
For bivariate , we will take 2 features as predictors , the remaining code , is the same .

```
X = df[['lot-area','taxvalue']] # predictor
y = df['retailvalue']
#y = df.iloc[:,0].values # target

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=0)
```

```
intercept_: 36138.6460388875
coef_: [9.34678258 1.04713042]

MAE:  7750.045887029295
MSE:  100277548.10319632
RMSE:  10013.067789300702
R^2 :  0.9055435226713242
Time:  0.016050815553227539
```

Plotting Bivariate Regression

$R^2 = 0.99$



Univariate Polynomial Regression

Degree 2

```
Degree  2
intercept_: 96568.61491507164
coef_: [ 0.00000000e+00  2.49296194e+02 -1.08273896e-02]

MAE:  21902
MSE:  809941866
RMSE:  28459
R^2 :  0.8832
Time:  0.023892879486083984
```

Degree 3

```
Degree  3
intercept_: 73270.14087444736
coef_: [ 0.00000000e+00  4.57150057e+02 -4.26391086e-01  2.28796000e-04]

MAE:  21185
MSE:  790506702
RMSE:  28117
R^2 :  0.8860
Time:  0.021661043167114258
```

Degree 4

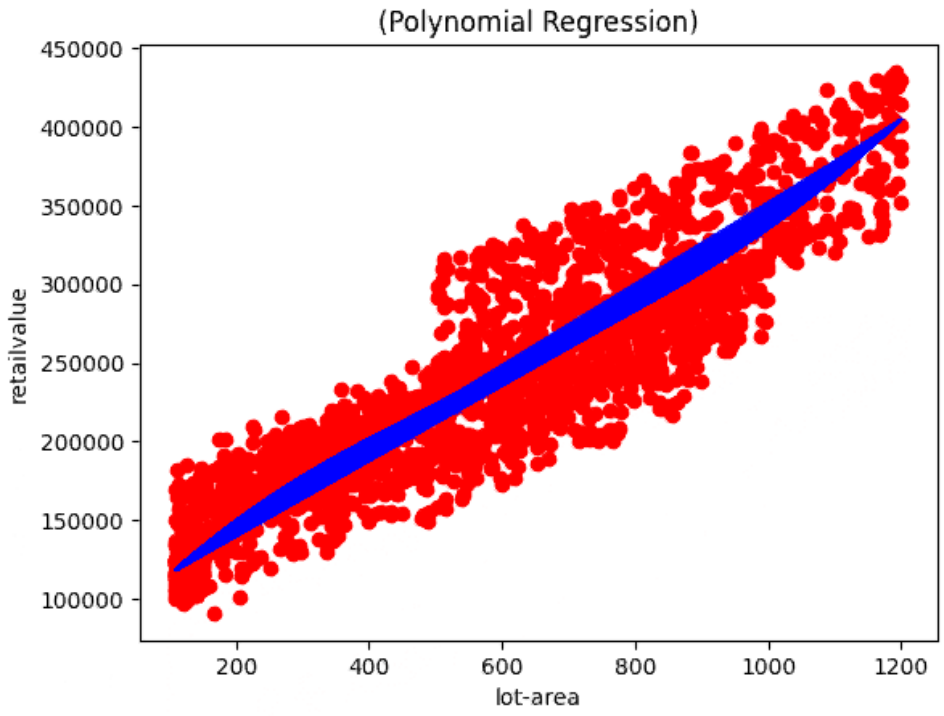
```
Degree  4
intercept_: 73359.29802560003
coef_: [ 0.00000000e+00  4.56121138e+02 -4.23085444e-01  2.24834096e-04
  1.57933434e-09]

MAE:  21185
MSE:  790616807
RMSE:  28118
R^2 :  0.8860
Time:  0.018103361129760742
```

Degree 5

```
Degree  5
intercept_: 75752.03195284595
coef_: [ 0.00000000e+00  4.22179993e+02 -2.69159377e-01 -7.29680731e-05
  2.58200765e-07 -8.09704586e-11]

MAE:  21195
MSE:  791269724
RMSE:  28130
R^2 :  0.8859
Time:  0.017685413360595703
```

I've tried untill degree 10 , in this case , the third and fourth had the same score , then the scores dropped slowly . The best is degree 3.

Plotting Univariate Polynomial Regression



Bivariate Polynomial Regression

Degree 2

```
Degree  2
intercept_: 43993.93153636533
coef_: [ 0.00000000e+00  8.43935814e+00  9.33635982e-01  5.98414773e-02
 -3.58846975e-04  8.51114425e-07]

MAE:  7682
MSE:  97807037
RMSE:  9890
R^2 :  0.9859
Time:  0.018610477447509766
```

Degree 3

```
Degree  3
intercept_: 122.14434605589486
coef_: [ 0.00000000e+00 -2.79292911e+02  2.49440858e+00 -2.40876042e-01
  4.32631926e-03 -1.40066541e-05  7.34741136e-05  5.58016408e-07
 -1.22162767e-08  3.52610033e-11]

MAE:  7509
MSE:  94201740
RMSE:  9706
R^2 :  0.9864
Time:  0.02237844467163086
```

Degree 4

```
Degree  4
intercept_: 74013.81075617182
coef_: [ 0.00000000e+00 -6.63762266e-06  3.78057675e-07 -5.50944204e-06
 -9.98135192e-04  1.27833565e-05 -6.49001540e-04  5.79766383e-06
 -6.55808069e-09 -5.22476369e-11 -6.14870651e-07  1.11259811e-08
 -6.54031247e-11  1.31959988e-13 -6.91261045e-18]

MAE:  7614
MSE:  96022552
RMSE:  9799
R^2 :  0.9062
Time:  0.024842262260066406
```

The same thing happened , it growed untill 0.9864 , then slowly dropped to 0.9862 and so on .

Logistic Regression

We will create a feature for Logistic Regression model , based on the retailvalue attribute . The new attribute will show the threshold between high-priced houses and low-priced houses , the treshold that is on the second quartile , 50% .

	retailvalue
count	2499.000000
mean	219054.021929
std	81895.336006
min	91000.000000
25%	137000.000000
50%	212000.000000
75%	282000.000000
max	435000.000000

Creating a feature for Logistic Regression

	retailvalue	zipcode	lot-len	lot-width	lot-area	house-area	garden-size	balcony	x-coor	y-coor	buildyear	bathrooms	taxvalue	energy-eff	monument	priceLogistic
0	232000	3520	35.1	14.2	499	147	406	1	2192	5470	1978	2	173100	0	0	1
1	184000	3520	22.2	18.2	404	109	328	1	2059	5228	1981	1	136000	0	1	0
2	230000	3520	59.7	11.8	704	89	629	1	2243	5388	1972	1	169000	0	0	1
3	257000	3520	39.0	18.6	726	130	669	1	2050	5432	1970	2	196900	0	0	1
4	286000	3520	44.2	18.1	800	160	750	0	2138	5110	1978	2	226000	0	0	1
5	185000	3520	16.8	11.4	192	154	124	0	2194	5214	1972	2	131200	0	0	0
6	182000	3520	6.7	16.5	111	159	19	0	2381	5378	1972	1	122100	0	0	0
7	184000	3520	17.4	14.4	250	139	180	1	2083	5292	1974	1	130900	0	0	0
8	266000	3520	48.8	17.4	849	115	796	1	2140	5192	1976	1	206400	0	0	1
9	280000	3520	76.3	11.6	885	118	793	0	2349	5240	1985	1	213600	0	1	1

Univariate

```
Intercept:  [-6.56699663]
Coef:  [[0.01327895]]
Result:  [1 0 1 ... 0 0 1]
Score:  0.9327731092436975
Time:  0.0164031982421875
confusion_matrix:
[[1146  110]
 [  58 1185]]
classification_report:
              precision    recall  f1-score   support

           0       0.95      0.91      0.93      1256
           1       0.92      0.95      0.93      1243

    accuracy                           0.93      2499
   macro avg       0.93      0.93      0.93      2499
weighted avg       0.93      0.93      0.93      2499
```

Bivariate

```
Intercept:  [-0.00012213]
Coef:  [[ 1.36935190e-02 -3.60889181e-05]]
Result:  [1 1 1 ... 1 0 1]
Score:  0.8339335734293718
Time:  0.015445947647094727
confusion_matrix:
[[ 996  260]
 [ 155 1088]]
classification_report:
              precision    recall  f1-score   support

           0       0.87      0.79      0.83      1256
           1       0.81      0.88      0.84      1243

    accuracy                           0.83      2499
   macro avg       0.84      0.83      0.83      2499
weighted avg       0.84      0.83      0.83      2499
```

Ridge, Lasso and ElasticNet

```
MSE:  805103008.21768
Time:  0.006439685821533203
```

```
0.8732353212041651
Time:  0.004304409027099609
```

```
Score eLastic :  0.8732353211831004
Time:  0.010235071182250977
```

KNN

```
MAE:  30992.0
MSE:  1818549333.3333333
RMSE:  42644.45255051744
R^2 :  0.7378294772289729
Time:  0.06321978569030762
```

SVC

Linear

```
Result:  [1 0 1 ... 0 0 1]
Score:  0.9295718287314926
Time:  0.015386343002319336
confusion_matrix:
[[1136  120]
 [  56 1187]]
classification_report:
              precision    recall  f1-score   support

           0       0.95      0.90      0.93      1256
           1       0.91      0.95      0.93      1243

    accuracy                           0.93      2499
   macro avg       0.93      0.93      0.93      2499
weighted avg       0.93      0.93      0.93      2499
```

Poly

```
Result:  [1 0 1 ... 1 0 1]
Score:  0.9211684673869548
Time:  0.14380574226379395
confusion_matrix:
[[1102  154]
 [  43 1200]]
classification_report:
              precision    recall  f1-score   support

           0       0.96      0.88      0.92      1256
           1       0.89      0.97      0.92      1243

    accuracy                           0.92      2499
   macro avg       0.92      0.92      0.92      2499
weighted avg       0.92      0.92      0.92      2499
```

radial basis function

```
Result:  [0 0 1 ... 0 0 1]
Score:  0.09195678271300 52
Time:  1.250969409942627
confusion_matrix:
[[1183   73]
 [ 197 1046]]
classification_report:
              precision    recall  f1-score   support

           0       0.86      0.94      0.90      1256
           1       0.93      0.84      0.89      1243

    accuracy                           0.89      2499
   macro avg       0.90      0.89      0.89      2499
weighted avg       0.90      0.89      0.89      2499
```

SVR

This dataset has 2499 rows and 16 columns , or features .

Linear

```
MAE:  49937.29615596176
MSE:  3348808474.35345
RMSE:  57868.8903846743
R^2 :  0.5172202082788571
Time:  0.01723623275756836
```

Poly

```
MAE:  26350.96539547016
MSE:  1128230517.6764596
RMSE:  33589.14285414946
R^2 :  0.8373490456355703
Time:  0.14835214614868164
```

radial basis function

```
MAE:  70437.32979032413
MSE:  7020078573.356591
RMSE:  83785.90915754624
R^2 :  -0.012047149744953023
Time:  0.21212506294250488
```

Nystroem/Classification-SVC

This dataset has 2499 rows and 16 columns , or features .

Linear

```
Result:  [0 0 1 ... 0 0 1]
Score:  0.9267707082833133
Time:  0.16744542121887207
confusion_matrix:
[[1158   98]
 [  85 1158]]
classification_report:
              precision    recall  f1-score   support

           0       0.93      0.92      0.93      1256
           1       0.92      0.93      0.93      1243

    accuracy                           0.93      2499
   macro avg       0.93      0.93      0.93      2499
weighted avg       0.93      0.93      0.93      2499
```

Poly

```
Result:  [1 1 1 ... 1 0 1]
Score:  0.7735094037615046
Time:  0.6935691833496094
confusion_matrix:
[[ 700  556]
 [  10 1233]]
classification_report:
              precision    recall  f1-score   support

           0       0.99      0.56      0.71      1256
           1       0.69      0.99      0.81      1243

    accuracy                           0.77      2499
   macro avg       0.84      0.77      0.76      2499
weighted avg       0.84      0.77      0.76      2499
```

radial basis function

```
Result:  [0 0 0 ... 0 0 0]
Score:  0.5026010404161665
Time:  3.486034870147705
confusion_matrix:
[[1256    0]
 [1243    0]]
classification_report:
              precision    recall  f1-score   support

           0       0.50      1.00      0.67      1256
           1       0.00      0.00      0.00      1243

    accuracy                           0.50      2499
   macro avg       0.25      0.50      0.33      2499
weighted avg       0.25      0.50      0.34      2499
```

Decision Trees

This dataset has 2499 rows and 16 columns , or features .

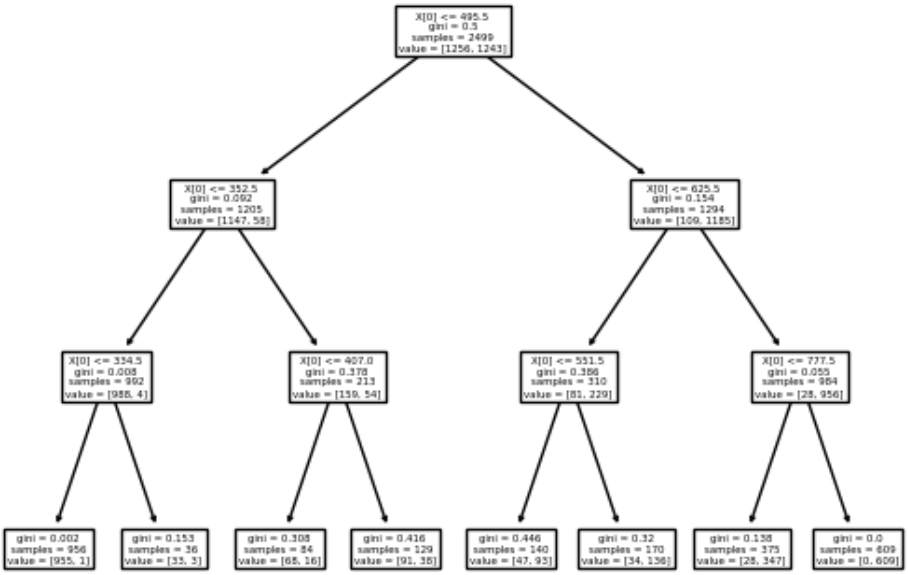
Regression

```
MAE:  27049.060589301764
MSE:  1405966591.9032333
RMSE:  37496.22103496875
R^2 :  0.7973093225234453
Time:  0.010043144226074219
```

Classification (with gini criterion)

```
Result:  [1 0 1 ... 0 0 1]
Score:  0.933173269307723
Time:  0.010600437030012695
confusion_matrix:
[[1147  109]
 [  58 1185]]
classification_report:
              precision    recall  f1-score   support

           0       0.95      0.91      0.93      1256
           1       0.92      0.95      0.93      1243

    accuracy                           0.93      2499
   macro avg       0.93      0.93      0.93      2499
weighted avg       0.93      0.93      0.93      2499
```

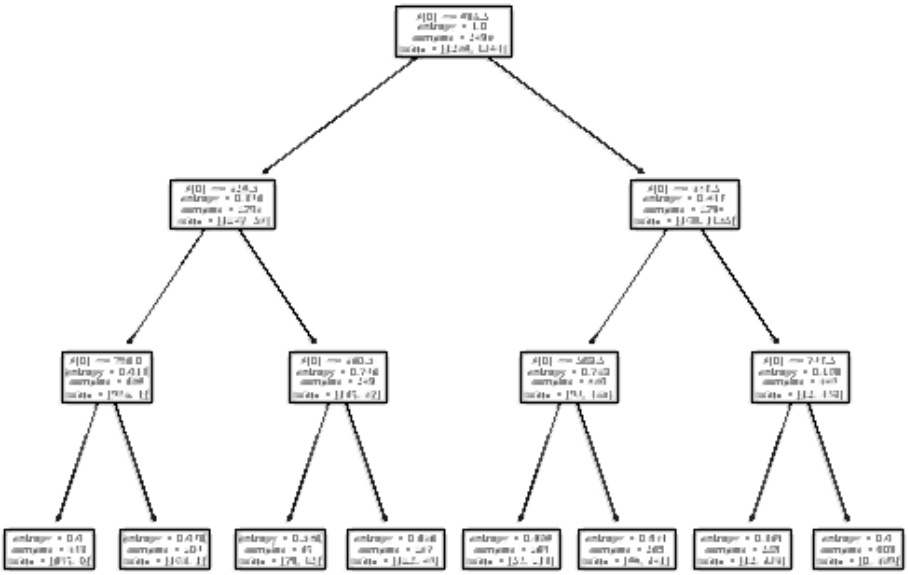
Tree Visualization



Classification (with entropy criterion)

```
Result:  [1 0 1 ... 0 0 1]
Score:  0.933173269307723
Time:  0.011295318603515625
confusion_matrix:
[[1147  109]
 [  58 1185]]
classification_report:
              precision    recall  f1-score   support

           0       0.95      0.91      0.93      1256
           1       0.92      0.95      0.93      1243

    accuracy                           0.93      2499
   macro avg       0.93      0.93      0.93      2499
weighted avg       0.93      0.93      0.93      2499
```

Tree Visualization



Random Forest

This dataset has 2499 rows and 16 columns , or features .

Regression

```
MAE:  25055.504743625657
MSE:  1146603231.4586546
RMSE:  33861.5302586675
R^2 :  0.8347003498379306
Time:  0.16644501686096191
```

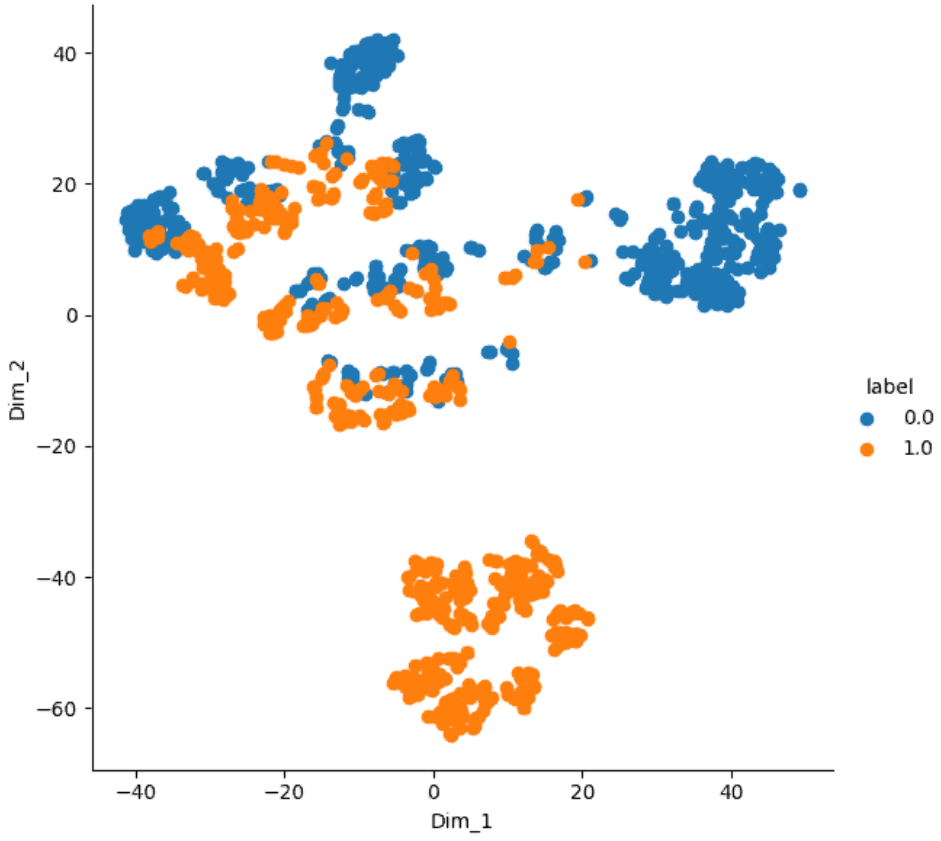
Classification

```
Result:  [1 0 1 ... 0 0 1]
Score:  0.952781112444978
Time:  0.3152439594268799
confusion_matrix:
[[1174   82]
 [  36 1207]]
classification_report:
              precision    recall  f1-score   support

           0       0.97      0.93      0.95      1256
           1       0.94      0.97      0.95      1243

    accuracy                           0.95      2499
   macro avg       0.95      0.95      0.95      2499
weighted avg       0.95      0.95      0.95      2499
```

TSNE

```
Result:  [1 0 1 ... 1 0 1]
Score:  0.9635854341736695
Time:  0.3649451732635498
confusion_matrix:
[[1192   64]
 [  27 1216]]
classification_report:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96      1256
           1       0.95      0.98      0.96      1243

    accuracy                           0.96      2499
   macro avg       0.96      0.96      0.96      2499
weighted avg       0.96      0.96      0.96      2499
```

TSNE Plot



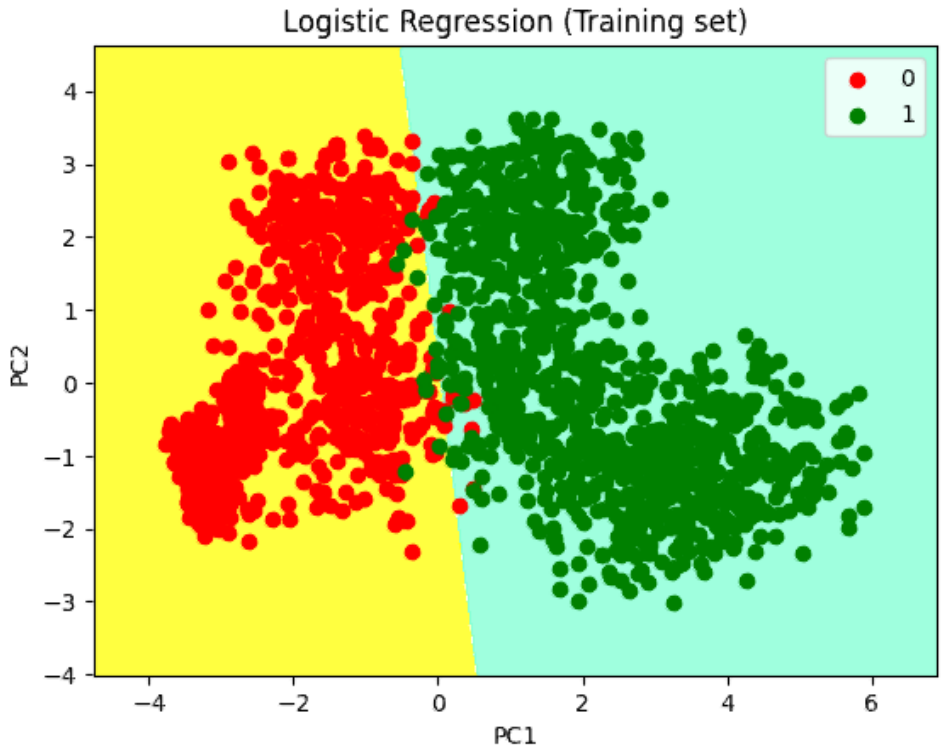
Kernel PCA

```
Result:  [1 0 1 ... 1 0 1]
Score:  0.9467787114845938
Time:  0.49223828315734863
confusion_matrix:
[[1161   95]
 [  38 1205]]
classification_report:
              precision    recall  f1-score   support

           0       0.97      0.92      0.95      1256
           1       0.93      0.97      0.95      1243

    accuracy                           0.95      2499
   macro avg       0.95      0.95      0.95      2499
weighted avg       0.95      0.95      0.95      2499
```

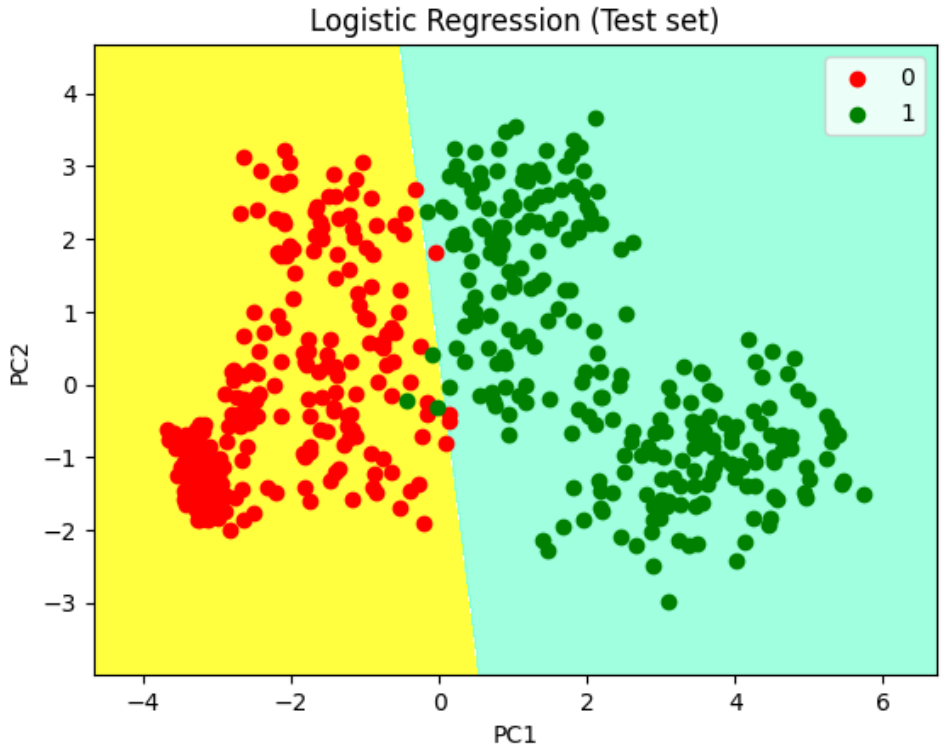
PCA - Logistic Regression

```
MAE:  0.012
MSE:  0.012
RMSE:  0.10954451150103323
R^2 :  0.9519992319877117
Time:  0.007677793502807617
```

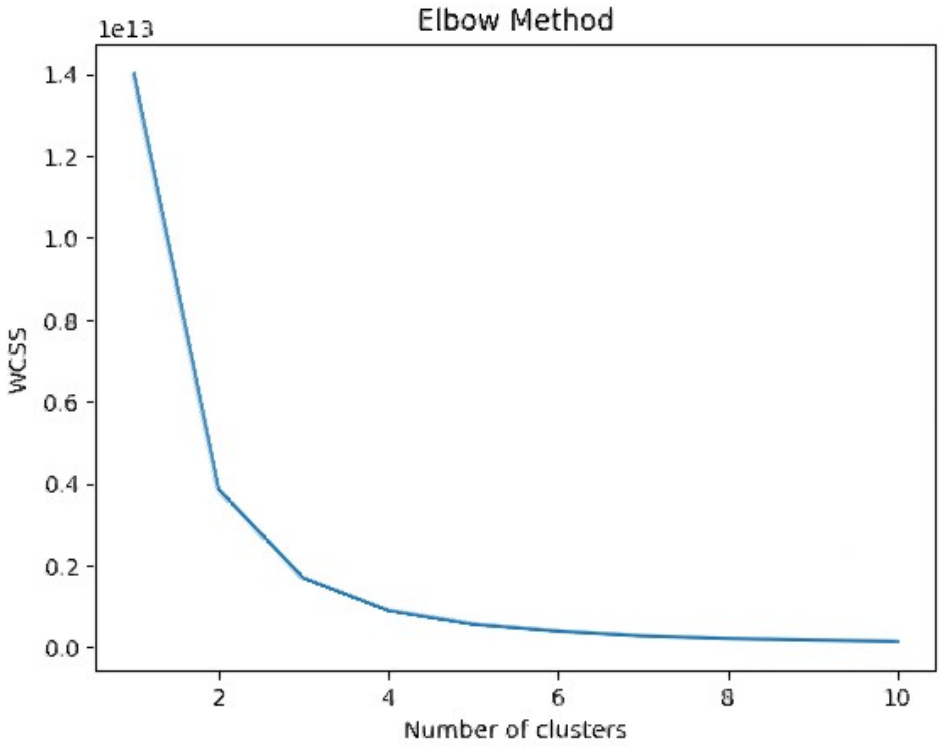
PCA Plot Training



PCA Plot Testing

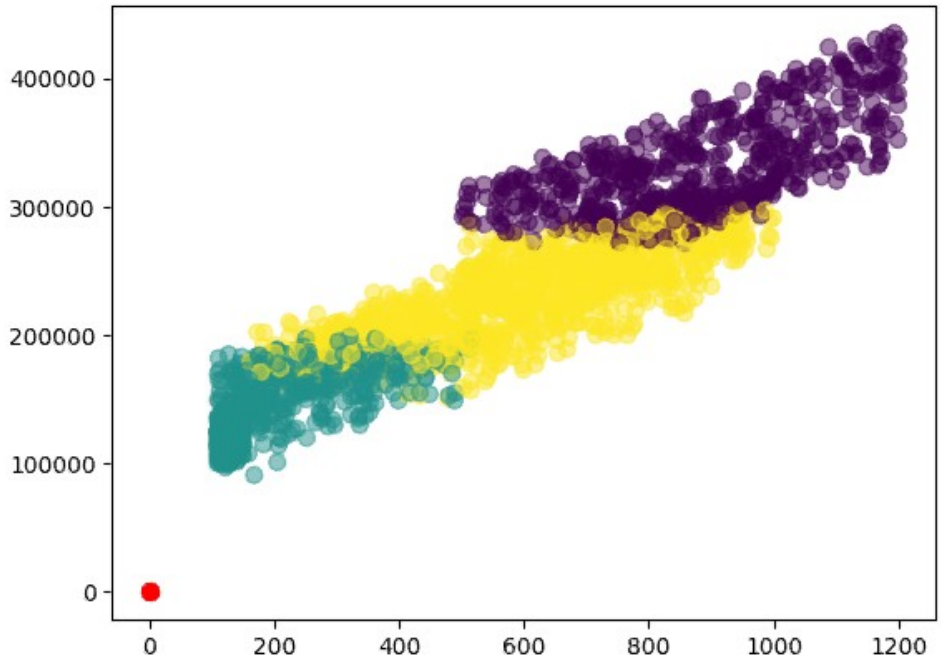


Elbow Method

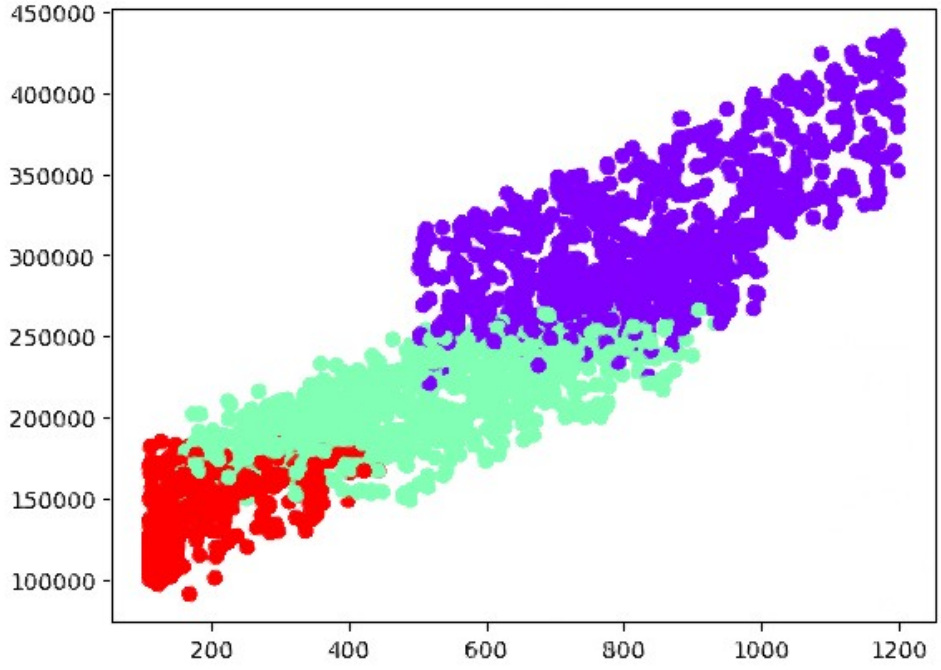


Clustering – lot-area and retailvalue

Kmeans

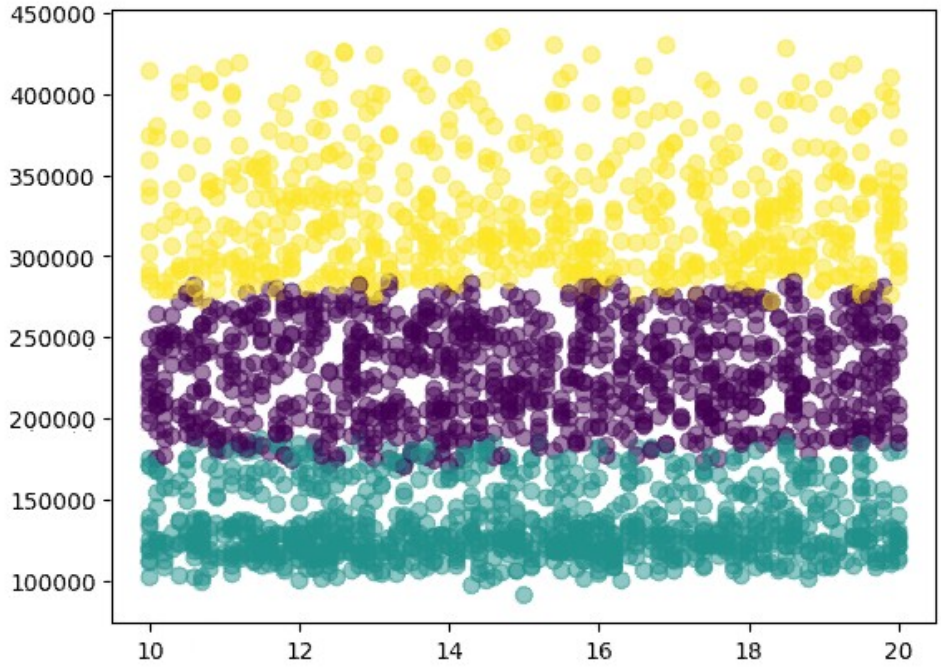


Agglomerative

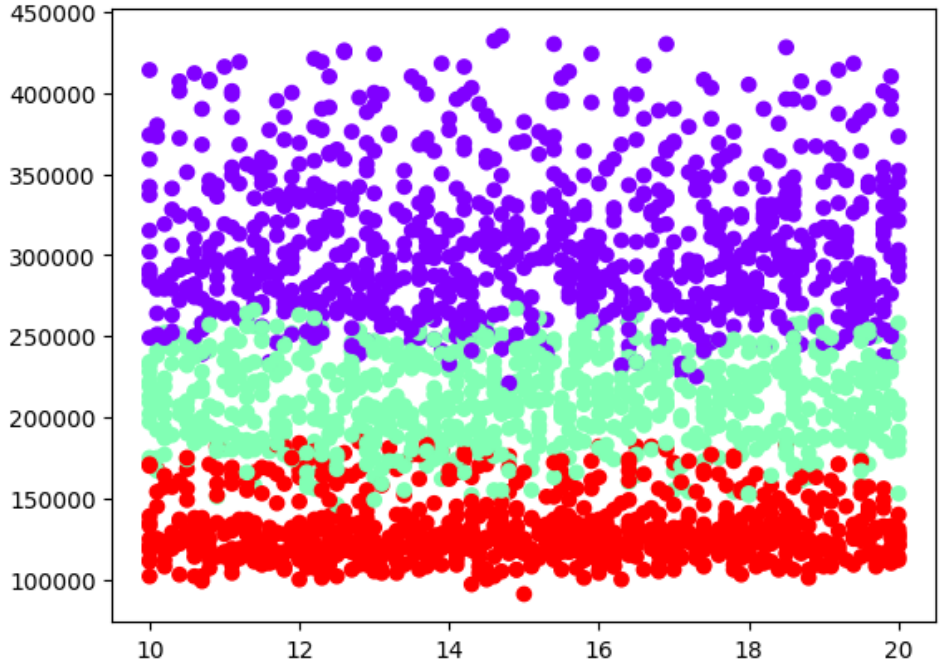


Clustering – lot-width and retailvalue

Kmeans



Agglomerative



Regression	Type	R^2	Cross-Validation	MEAN (Cross-Validation)	TIME
Univariate Linear Regression		0.8839326089985096			0.01648259162902832
Bivariate Linear Regression		0.9855435226713242			0.01333308219909668
Univariate Polynomial Regression	Degree-2	0.8832			0.0176999568939209
	Degree-3	0.8860			0.023972272872924805
Bivariate Polynomial Regression	Degree-2	0.9859			0.011221885681152344
	Degree-3	0.9864			0.017053842544555664
Logistic Regression	Univariate	0.9327731092436975			0.013450145721435547
	Bivariate	0.8339335734293718			0.018367290496826172
Ridge	Not a type MSE:	805103008.21768			0.004631757736206055
Lasso		0.8732353212041651			0.006082296371459961
ElasticNet		0.8732353211831004			0.008506536483764648
K-nearest neighbors		0.7378294772289729	r2: [0.73523295 0.73816774 0.7665821 0.74091509	0.741531751115086	0.030174493778967285

			0.72676087]		
SVC	linear	0.9295718287314926	r2: [0.664 0.67997952 0.77594264 0.75999616 0.73541888]	0.7230674391743643	0.015743494033813477
	poly	0.9211684673869548	r2: [0.664 0.63997696 0.76794059 0.67199475 0.67929561]	0.6846415820836903	0.12989568710327148
	rbf	0.8919567827130852	r2: [-0.112 0.04793907 -0.05627041 0.0559849 0.01383399]	-0.010102489854037078	0.8501145839691162
SVR	linear	0.517353181357521	r2: [0.43113634 0.43754391 0.47957509 0.52916761 0.47007231]	0.4694990537855683	0.013514280319213867
	poly	0.8373490456355703	r2: [0.8128313 0.7818608 0.82083026 0.82747687 0.81741196]	0.8120822381083886	0.15265512466430664
	rbf	-0.012047149744953023	r2: [-0.00902752 -0.01354 -0.00143056 -0.02130689 -0.00728011]	-0.010517013641642814	0.1808793544769287

Nystroem	linear	0.9267707082833133	r2: [0.664 0.67997952 0.77594264 0.75999616 0.73541888]	0.7230674391743643	0.1675586700439453
	poly	0.7739095638255302	r2: [0.664 0.63997696 0.76794059 0.67199475 0.67929561]	0.7110503464557106	0.6729810237884521
	rbf	0.5026010404161665	r2: [-0.112 0.04793907 -0.05627041 0.0559849 0.01383399]	-0.010102489854037078	1.9959745407104492
Decision Trees	Regression	0.7973093225234453	r2: [0.80280485 0.79619226 0.83031905 0.81129357 0.8015902]	0.8084399862 03505	0.0211586952 20947266

Decision Trees	Classification gini	0.9331732693 07723	r2: [0.672 0.68798003 0.75993854 0.79199667 0.73541888]	0.7294668245 727879	0.0100448131 5612793
	Classification entropy	0.9331732693 07723	r2: [0.672 0.68798003 0.77594264 0.79199667 0.73541888]	0.7326676439 825568	0.0098295211 79199219

Random Forest	Regression	0.8347003498379306	r2: [0.84668132 0.81888612 0.85988378 0.86046315 0.83306976]	0.8437968261410418	0.1360030174255371
Random Forest	Classification	0.9531812725090036	r2: [0.584 0.61597542 0.7039242 0.68799501 0.71136605]	0.6606521362820655	0.2874407768 2495117

Logistic Regression	PCA	0.9519992319877117	r2: [0.864 0.87999232 0.92798156 0.95199923 0.89577107]	0.9039488374223179	0.0123882293 70117188

Random Forest	Kernel PCA	0.9463785514205683	r2: [0.32 0.30395545 0.51987709 0.44799117 0.39066165]	0.3964970726388923	0.4568672180 1757810.9864

Random Forest	TSNE	0.9635854341736695	r2: [0.608 0.63197645 0.68792011 0.63999424 0.65524278]	0.6446267143344332	0.3213977813720703

Univariate Polynomial Regression Degree 4-10	0.8860	0.8859	0.8841	0.8812	0.8765	0.8700	0.8620
Bivariate Polynomial Regression Degree 4-10	0.9862	0.9846	0.9776	0.9610	0.9344	0.9001	0.8611

The best classifier is Bivariate Polynomial Regression with Degree 3 , and at the other end , is SVC with radial basis function and Nystroem kernel aproximation . Behind first classifier is Bivariate Polynomial Regression with Degree 2 , and better than the last is SVR with linear .