Module: Applied Statistics & Machine Learning (CA_TWO)

Submitted by: Kunal Bhoite

Student ID: 20000769

Lecturer Name: Dr. Assem Abdelhak

Q1) Impact of L1, L2, and elastic net regularization on linear regression coefficients, performance, and interpretability. (Note - models built with fewer variables are considered more interpretable)

Answer-

In Linear Regression we avoid overfitting using mainly two methods Feature Reduction and Regularization. Feature Reduction can be done manually or by Correlation Heatmap. Removing the features manually means identifying which is the least important feature and dropping that feature. A Correlation Heatmap is used to understand which features are related to each other and the strength between each feature. In Correlation Heatmap there are two factors Correlation and Causation. Correlation is the measure that expresses the strength between two features. Mainly the value of Correlation lies between -1 to 1. Causation is the relationship between two variables, Causation basically means the dependency of one variable on another variable. If there is strong causation means if one variable is dependent on another variable so we can drop any one of them. In this way using Correlated Heat map we can reduce the number of features and can only keep the important features.

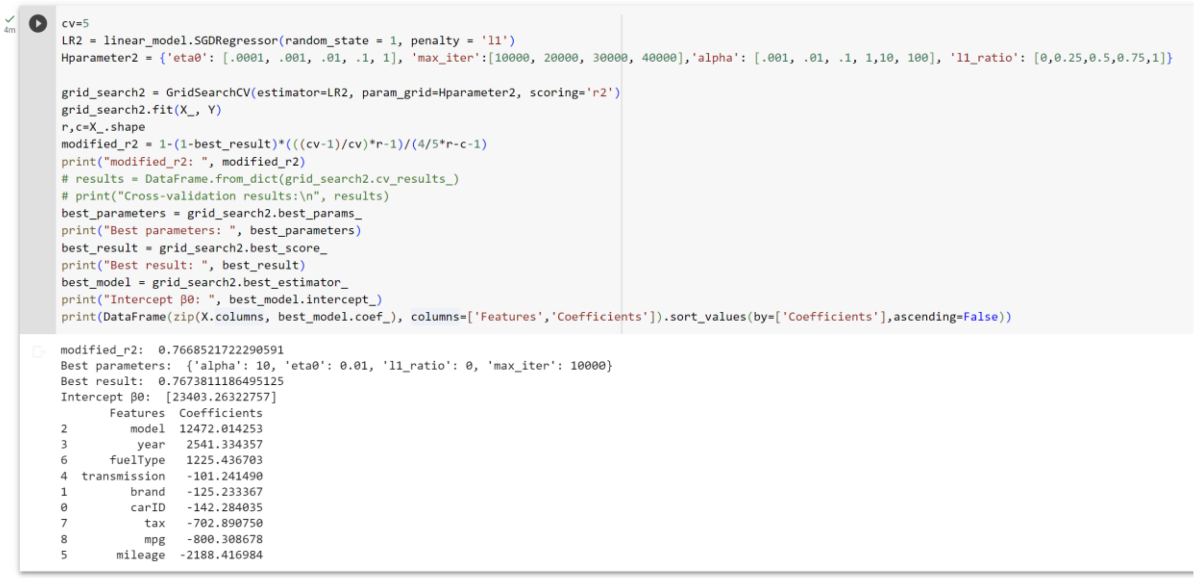
The Second method is through regularization which means adding penalties to the Cost Function.

There are 3 types of Regularization.

1. **Lasso Regularization(L1)**

   a) L1 regularization adds the absolute values of the coefficients to the loss function.
   The regularization term is directly proportional to the sum of the absolute values of the coefficients.
   loss function of L1 is as follows-
   Loss = Mean Squared Error + λ * ∑|βi|
   λ = regularization strength hyperparameter
   βi = coefficients

   b) **Performance:**



```
cv=5
LR2 = linear_model.SGDRegressor(random_state = 1, penalty = 'l1')
Hparameter2 = {'eta0': [.0001, .001, .01, .1, 1], 'max_iter':[10000, 20000, 30000, 40000],'alpha': [.001, .01, .1, 1,10, 100], 'l1_ratio': [0,0.25,0.5,0.75,1]}

grid_search2 = GridSearchCV(estimator=LR2, param_grid=Hparameter2, scoring='r2')
grid_search2.fit(X_, Y)
r,c=X_.shape
modified_r2 = 1-(1-best_result)*(((cv-1)/cv)*r-1)/(4/5*r-c-1)
print("modified_r2: ", modified_r2)
# results = DataFrame.from_dict(grid_search2.cv_results_)
# print("Cross-validation results:\n", results)
best_parameters = grid_search2.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search2.best_score_
print("Best result: ", best_result)
best_model = grid_search2.best_estimator_
print("Intercept β0: ", best_model.intercept_)
print(DataFrame(zip(X.columns, best_model.coef_), columns=['Features','Coefficients']).sort_values(by=['Coefficients'],ascending=False))
```

```
modified_r2:  0.7668521722290591
Best parameters:  {'alpha': 10, 'eta0': 0.01, 'l1_ratio': 0, 'max_iter': 10000}
Best result:  0.7673811186495125
Intercept β0:  [23403.26322757]
       Features   Coefficients
2         model  12472.014253
3          year   2541.334357
6      fuelType   1225.436703
4  transmission   -101.241490
1         brand   -125.233367
0         carID   -142.284035
7           tax   -702.890750
8           mpg   -800.308678
5       mileage  -2188.416984
```

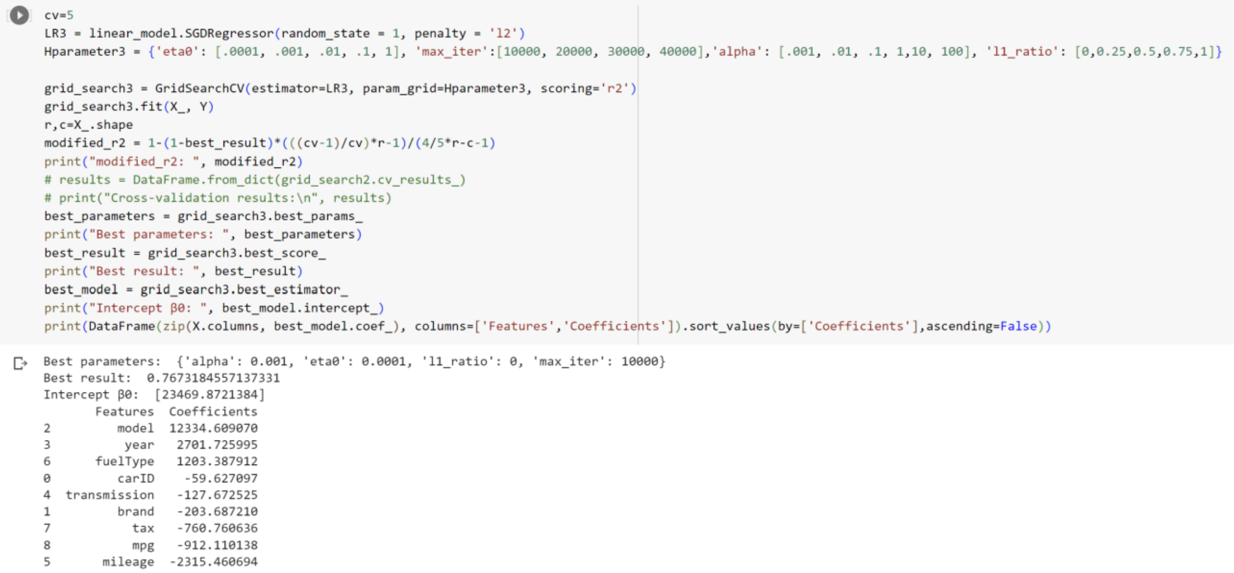
In our given dataset using L1 regularization performance is optimum when value of alpha is 10,value of eta is 0.01 L1 ratio is 0,and max iteration is 10000.
The Best r2 scored achieved by L1 regularization is 76.68%
L1 focuses on more relevant features ,and ignores irrelevant features thus it is more effective on high dimension dataset.

c) **Interpretability:**
For feature selection,Lasso is advantageous as our dataset has restricted predictions capabilities.
Avoids Irrelevant features and focuses on relevant features.

2. **Ridge Regularization(L2)**

a) L2 regularization adds the squared values of the coefficients to the loss function.
The regularization term is directly proportional to the sum of the squared values of the coefficients.
Loss function of L2 is as follows-
Loss = Mean Squared Error + λ * ∑(βi^2)
λ = regularization strength hyperparameter
βi = coefficients

b) **Performance:**



```
cv=5
LR3 = linear_model.SGDRegressor(random_state = 1, penalty = 'l2')
Hparameter3 = {'eta0': [.0001, .001, .01, .1, 1], 'max_iter':[10000, 20000, 30000, 40000],'alpha': [.001, .01, .1, 1,10, 100], 'l1_ratio': [0,0.25,0.5,0.75,1]}

grid_search3 = GridSearchCV(estimator=LR3, param_grid=Hparameter3, scoring='r2')
grid_search3.fit(X_, Y)
r,c=X_.shape
modified_r2 = 1-(1-best_result)*(((cv-1)/cv)*r-1)/(4/5*r-c-1)
print("modified_r2: ", modified_r2)
# results = DataFrame.from_dict(grid_search2.cv_results_)
# print("Cross-validation results:\n", results)
best_parameters = grid_search3.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search3.best_score_
print("Best result: ", best_result)
best_model = grid_search3.best_estimator_
print("Intercept β0: ", best_model.intercept_)
print(DataFrame(zip(X.columns, best_model.coef_), columns=['Features','Coefficients']).sort_values(by=['Coefficients'],ascending=False))
```

```
Best parameters:  {'alpha': 0.001, 'eta0': 0.0001, 'l1_ratio': 0, 'max_iter': 10000}
Best result:  0.7673184557137331
Intercept β0:  [23469.8721384]
       Features  Coefficients
2         model  12334.609070
3          year   2701.725995
6      fuelType   1203.387912
0         carID    -59.627097
4  transmission   -127.672525
1         brand   -203.687210
7           tax   -760.760636
8           mpg   -912.110138
5       mileage  -2315.460694
```

In our given dataset using L2 regularization performance is optimum when value of alpha is 0.001,value of eta is 0.0001 L1 ratio is 0,and max iteration is 10000.
The Best r2 scored achieved by L1 regularization is 76.68%
Handles collinearity.
Adds squared values to the coefficient.
Multi collinear means when two features are highly co related to each other.
Eg:Feature 1 is garage area and Feature 2 is Total area above ground
So these two features are highly correlated to each other.

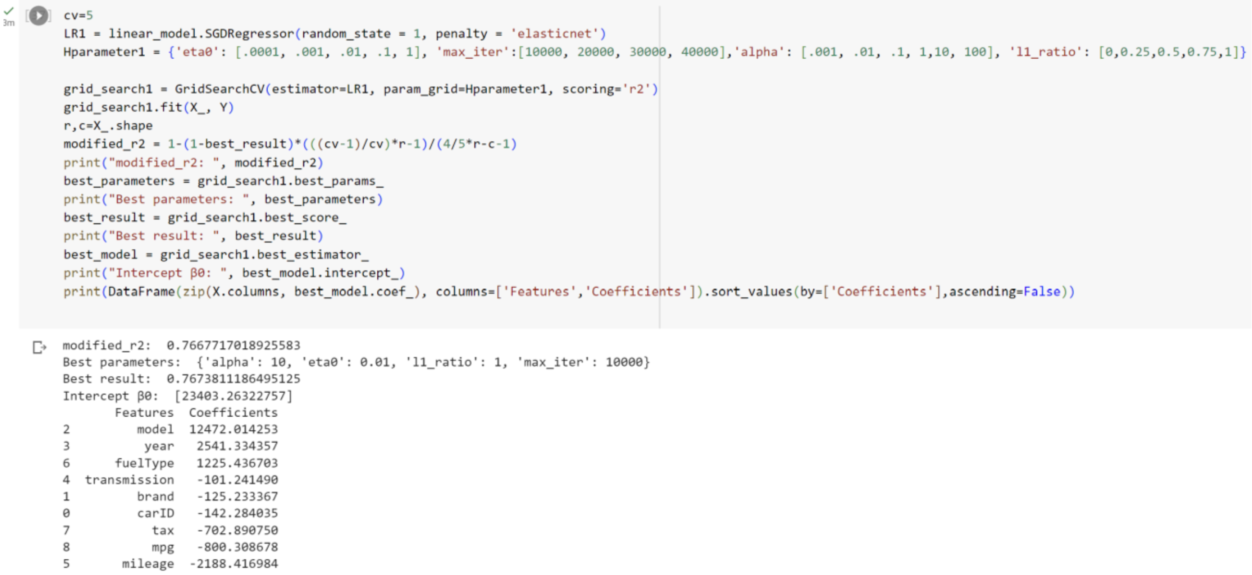
c) **Interpretability:**
It does not focus on feature selection unlike L1,which makes it less interpretable to high dimension data.
Ridge regularization is useful for the grouping effect since it allow selection of co-linear characteristics.

3. **Elastic Net**

a) Elastic Net regularization is combination of both L1 and L2 regularization.
It adds both the absolute values and the squared values of the coefficients to the loss function.

Loss function of Elastic Net is as follows-
min σ*i*=1 *m β*0+ σ*j*=1 *n βj xj i* − *y i* 2 + α(*λ* σ*j*=1 *n βj* + (1 − *λ*) σ*j*=1 *n βj* 2)
0 ≤ *λ* ≤ 1
λ is the regularization strength hyperparameters for L1 and L2 penalties respectively
βi = coefficients

b) Performance:



```
cv=5
LR1 = linear_model.SGDRegressor(random_state = 1, penalty = 'elasticnet')
Hparameter1 = {'eta0': [.0001, .001, .01, .1, 1], 'max_iter':[10000, 20000, 30000, 40000],'alpha': [.001, .01, .1, 1,10, 100], 'l1_ratio': [0,0.25,0.5,0.75,1]}

grid_search1 = GridSearchCV(estimator=LR1, param_grid=Hparameter1, scoring='r2')
grid_search1.fit(X_, Y)
r,c=X_.shape
modified_r2 = 1-(1-best_result)*(((cv-1)/cv)*r-1)/(4/5*r-c-1)
print("modified_r2: ", modified_r2)
best_parameters = grid_search1.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search1.best_score_
print("Best result: ", best_result)
best_model = grid_search1.best_estimator_
print("Intercept β0: ", best_model.intercept_)
print(DataFrame(zip(X.columns, best_model.coef_), columns=['Features','Coefficients']).sort_values(by=['Coefficients'],ascending=False))
```

```
modified_r2:  0.7667717018925583
Best parameters:  {'alpha': 10, 'eta0': 0.01, 'l1_ratio': 1, 'max_iter': 10000}
Best result:  0.7673811186495125
Intercept β0:  [23403.26322757]
       Features  Coefficients
2         model  12472.014253
3          year   2541.334357
6      fuelType   1225.436703
4  transmission   -101.241490
1         brand   -125.233367
0         carID   -142.284035
7           tax   -702.890750
8           mpg   -800.308678
5       mileage  -2188.416984
```

In our given dataset using Elastic Net regularization performance is optimum when value of alpha is 10,value of eta is 0.01 L1 ratio is 1,and max iteration is 10000.
The Best r2 scored achieved by L1 regularization is 76.68%

c) Interpretability:
It is the combination of both L1 and L2 Regularization. It adds both squared values as well as absolute values to the co-efficient. If the value of *λ*=1 then it be L1 regularization if *λ*=0 then it will be L2 regularization. If *λ*=0.5 then it will use 50% L1 and 50% L2.
Interpretability of model is higher as it combined strength of both Lasso regularization and Ridge regularization.It uses feature selection of L1 and Co-linearity of L2.

Q2) Impact of L2 regularization on support vector regression performance and interpretability.

Answer- L2 regularization is a technique used to prevent overfitting in machine learning models, including Support Vector Regression (SVR). It achieves this by adding a penalty term to the model's cost function, discouraging large coefficient values. The impact of L2 regularization on SVR performance and interpretability can be summarized as follows.

Performance:

a) L2 regularization helps improve SVR performance by reducing overfitting.
b) leading to better generalization on unseen data.
c) It effectively limits the model's complexity.

d) Preventing it from fitting noise in the training data and ensuring a smoother decision boundary.

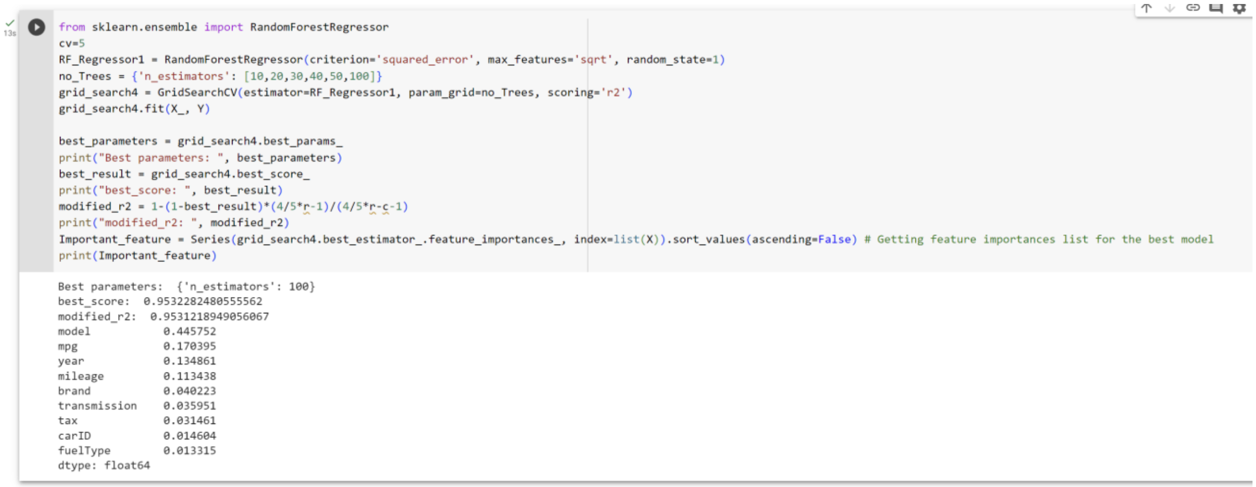
Interpretability:

a) L2 regularization promotes sparsity in the SVR model, meaning it encourages some feature coefficients to become exactly zero.
b) This leads to a more interpretable model as it identifies and prioritizes the most relevant features, simplifying the understanding of the relationship between input features and the target variable.

Moreover, the regularization term helps in feature selection, discarding less important features, and making the model easier to interpret by practitioners and domain experts.

Q3)If you were to implement random forest regression, then its comparative performance and interpretability with respect to regularized linear regression and regularized support vector regression models.

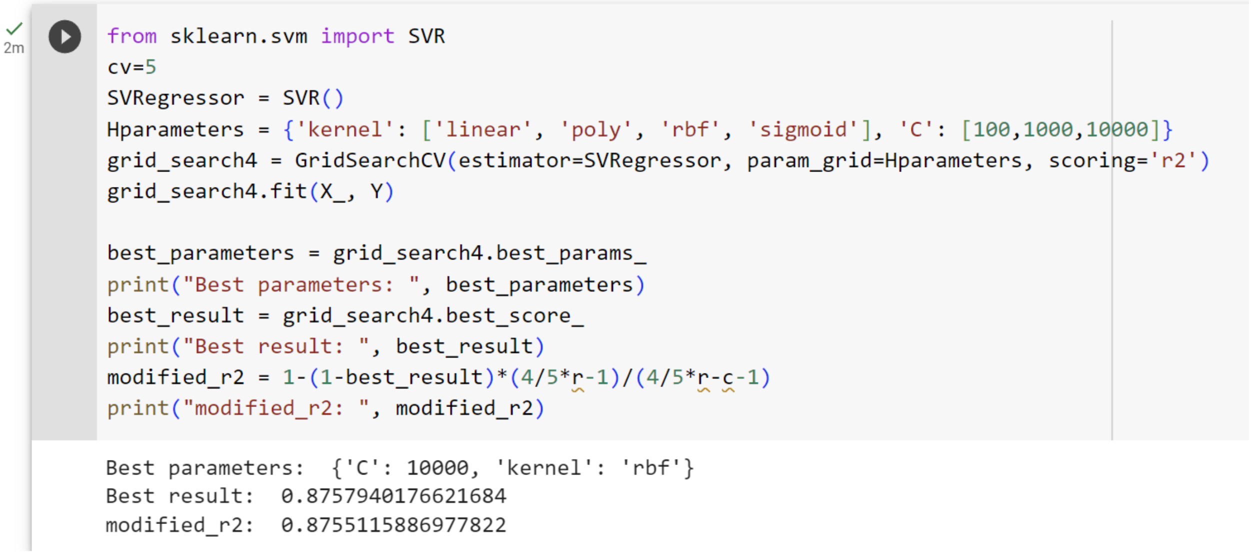
Answer-



```
from sklearn.ensemble import RandomForestRegressor
cv=5
RF_Regressor1 = RandomForestRegressor(criterion='squared_error', max_features='sqrt', random_state=1)
no_Trees = {'n_estimators': [10,20,30,40,50,100]}
grid_search4 = GridSearchCV(estimator=RF_Regressor1, param_grid=no_Trees, scoring='r2')
grid_search4.fit(X_, Y)

best_parameters = grid_search4.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search4.best_score_
print("best_score: ", best_result)
modified_r2 = 1-(1-best_result)*(4/5*r-1)/(4/5*r-c-1)
print("modified_r2: ", modified_r2)
Important_feature = Series(grid_search4.best_estimator_.feature_importances_, index=list(X)).sort_values(ascending=False) # Getting feature importances list for the best model
print(Important_feature)
```

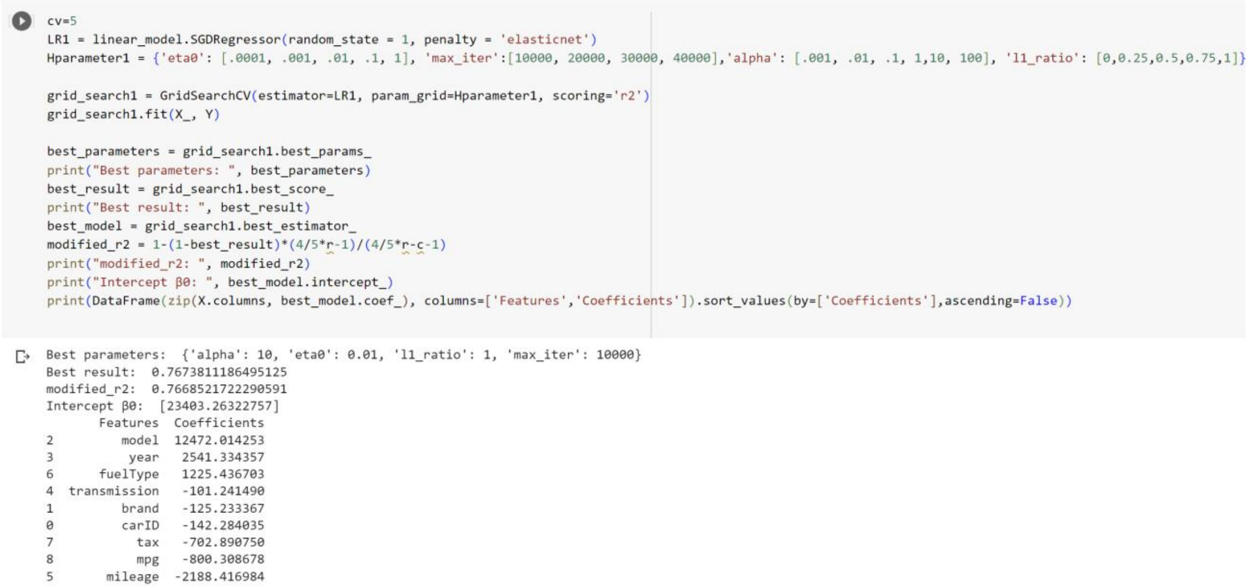
```
Best parameters:  {'n_estimators': 100}
best_score:  0.9532282480555562
modified_r2:  0.9531218949056067
model           0.445752
mpg             0.170395
year            0.134861
mileage         0.113438
brand           0.040223
transmission    0.035951
tax             0.031461
carID           0.014604
fuelType        0.013315
dtype: float64
```



```
from sklearn.svm import SVR
cv=5
SVRegressor = SVR()
Hparameters = {'kernel': ['linear', 'poly', 'rbf', 'sigmoid'], 'C': [100,1000,10000]}
grid_search4 = GridSearchCV(estimator=SVRegressor, param_grid=Hparameters, scoring='r2')
grid_search4.fit(X_, Y)

best_parameters = grid_search4.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search4.best_score_
print("Best result: ", best_result)
modified_r2 = 1-(1-best_result)*(4/5*r-1)/(4/5*r-c-1)
print("modified_r2: ", modified_r2)
```

```
Best parameters:  {'C': 10000, 'kernel': 'rbf'}
Best result:  0.8757940176621684
modified_r2:  0.8755115886977822
```



```
cv=5
LR1 = linear_model.SGDRegressor(random_state = 1, penalty = 'elasticnet')
Hparameter1 = {'eta0': [.0001, .001, .01, .1, 1], 'max_iter':[10000, 20000, 30000, 40000],'alpha': [.001, .01, .1, 1,10, 100], 'l1_ratio': [0,0.25,0.5,0.75,1]}

grid_search1 = GridSearchCV(estimator=LR1, param_grid=Hparameter1, scoring='r2')
grid_search1.fit(X_, Y)

best_parameters = grid_search1.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search1.best_score_
print("Best result: ", best_result)
best_model = grid_search1.best_estimator_
modified_r2 = 1-(1-best_result)*(4/5*r-1)/(4/5*r-c-1)
print("modified_r2: ", modified_r2)
print("Intercept β0: ", best_model.intercept_)
print(DataFrame(zip(X.columns, best_model.coef_), columns=['Features','Coefficients']).sort_values(by=['Coefficients'],ascending=False))
```

```
Best parameters:  {'alpha': 10, 'eta0': 0.01, 'l1_ratio': 1, 'max_iter': 10000}
Best result:  0.7673811186495125
modified_r2:  0.7668521722290591
Intercept β0:  [23403.26322757]
       Features  Coefficients
2         model  12472.014253
3          year   2541.334357
6      fuelType   1225.436703
4  transmission   -101.241490
1         brand   -125.233367
0         carID   -142.284035
7           tax   -702.890750
8           mpg   -800.308678
5       mileage  -2188.416984
```

a) Performance:

In the first screenshot we can see the we have implemented Random Forest Regression, Random Forest Regression shows best result for n_estimator=100, which is the number of trees. The modified r2 and the best score both are similar so we can say that r2 score is 95.31%.

As compared to the Support Vector Regression in screenshot 2 we can see that it shows the optimal score when the kernel is ‘rbf’ and when c=10000.It shows r2 score as 87.57%.This score is also good because the modified r2 is also the same as the calculated r2.

As compared to the Regularized Linear Regression in screenshot 3 we can see that it shows optimal score when alpha=10,eta=0.01,L1_RATIO=1,max_iterations=10000.It shows r2 score as 76.68%. This score is also good because the modified r2 is also the same as the calculated r2.

In conclusion comparing Random Forest Regression ,Support Vector Regression,Regularized Linear Regression we can say that Random Forest Regression shows best score for our given dataset which is

95.31%.If we would have implemented Random Forest Regression for our given data set it would have predicted the car prices more accurately as compared to Support Vector Regression and Regularized Linear Regression. We can also say that our given dataset is not linear

b) Interpretability:

Random Forest Regression works good on Non Linear dataset and complex dataset. Support Vector Regression works good on Non Linear dataset. Regularized Linear Regression works good on Linear Dataset. As we can see for our given dataset the Regularized Linear Regression and Support Vector Regression are able to recognize the pattern accurately. So we can say that our dataset is Non Linear in nature. Hence to confirm our result we tried using Random Forest Regression and it showed the best possible result which is 95.31%.

In conclusion, the choice of the regression model (Random Forest Regression, regularized linear regression, or regularized SVR) depends on the data complexity and the relationship between features and the target variable. For the given dataset, the Random Forest Regressor obtained the highest score of 95.32% among all the regressor models, suggesting that it is well suited for this dataset.