ADVANCED ML FEATURE SELECTION

Advanced ML Feature Selection has two types

- **Feature Selection Method**
- **Dimensionality Reduction**

Feature Selection Method:

Feature selection in machine learning is the process of identifying and selecting the most relevant features (variables or attributes) from a dataset to build a more accurate and efficient model. It has following algorithm

a) SelectK Algorithm
b) Recursive Feature Elimination
c) Feature Importance
d) Forward Selection
e) Backward Selection

SelectK Algorithm

It support both regression and classification. Using selectkbest method we can find out best features. This method contains **chi square** and **RMS** value. We can use either chi square or RMS value to evaluate best feature in our dataset.

Coding:

```
#Create a function selectkBest with 3 parameter
 def selectkbest(indep_x,dep_y,n):
#select best feature using chi squared
test = SelectKBest(score_func=chi2, k=n)
#create a model based on best feature
fit1=test.fit(indep_x,dep_y
#Create a new indep_x dataset based on k values
selectk_features= fit1.transform(indep_x)
```

Output:

SelectK Algorithm for Classification

K=5

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
ChiSquare	0.94	0.94	0.95	0.89	0.83	0.96	0.95

Decision Tree, Random Forest and SVMnl perform very well.Navie Bays performs worst in this case.

SelectK Algorithm for Regression

K=5

	Linear	SVMl	SVMnl	Decision	Random
ChiSquare	0.551985	0.545395	0.749654	0.696181	0.836806

In this case Random Forest, SVMnl doing very well. At the same time Linear Regression, SVMl performs worst.

Based on the output (Regression &Classification), Classification algorithm is best for feature selection.

Recursive Feature Elimination:

Recursive Feature Elimination (RFE) is a feature selection technique used in machine learning to identify the most relevant features in a dataset by iteratively removing the least important ones.

In this method we should pass either **Regression or Classification Model** as a argument to find out best feature.

Coding:

```
#Function with 3 arguments
def rfeFeature(indep_X,dep_Y,n):
 #create a empty list
 rfelist=[]
```

```
#Create model
log_model = LogisticRegression(solver='lbfgs')
RF=RandomForestClassifier(n_estimators=10,criterion=    'entropy',
random_state = 0)
DT=DecisionTreeClassifier(criterion='gini',max_features='sqrt',
splitter='best',random_state = 0)
svc_model = SVC(kernel = 'linear', random_state = 0)
# create a list with  4 model
rfemodellist=[log_model,RF,DT,svc_model]
        # i is variable that choose each value from rfemodellist
        for i in   rfemodellist:
                print(i)
                #call RFE method
                log_rfe= RFE(estimator=i,n_features_to_select=n)
               #Create model
               log_fit = log_rfe.fit(indep_X, dep_Y)
               log_rfe_feature=log_fit.transform(indep_X)
               #Append selected feature in a rfelist
              rfelist.append(log_rfe_feature)
          return  rfelist
```

Output:

RFE Algorithm for Classification

n=3

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
Logistic	0.94	0.94	0.94	0.94	0.94	0.94	0.94
SVC	0.94	0.94	0.94	0.94	0.9	0.91	0.92
Random	0.98	0.98	0.98	0.98	0.79	0.97	0.97
DecisionTree	0.87	0.87	0.87	0.87	0.87	0.87	0.87

Best Performance:

Random forest consistently gives 0.98 accuracy with other models expect Navie Bays & Decision Tree.

Lowest Performance:

Navie Bays performs poorly with Random Forest and SVC.

In Decision Tree all values are 0.87

n=5

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
Logistic	0.98	0.98	0.98	0.98	0.98	0.98	0.98
SVC	0.97	0.97	0.98	0.97	0.91	0.96	0.98
Random	0.95	0.98	0.93	0.94	0.85	0.97	0.98
DecisionTree	0.99	0.99	0.99	0.99	0.99	0.99	0.99

Best Performance:

Decision Tree consistently gives 0.99 accuracy with other models Logistic Regression consistently gives 0.98 accuracy.

Lowest Performance:

Navie Bays performs poorly with Random Forest and svc.

Selecting 5 features with RFE gives better performances across all the models than selecting only 3 features.

RFE Algorithm for Regression

n=3

	Linear	SVMl	Decision	Random
Linear	0.441961	0.262153	0.441961	0.441816
SVRl	0.441961	0.262153	0.441961	0.441816
Random	0.664893	0.609652	0.965961	0.916304
DecisionTree	0.676174	0.670539	0.933504	0.887256

Best Performance:

Decision Tree (0.93) and Random Forest (0.96) gives best performance using random selected features.

Lowest Performance:

SVM Linear (0.26) consistently performed poorly.

n=5

	Linear	SVMl	Decision	Random
Linear	0.620124	0.457136	0.77924	0.780135
SVRl	0.604508	0.456848	0.776474	0.776745
Random	0.674403	0.628206	0.696181	0.815538
DecisionTree	0.686361	0.64332	0.836806	0.845303

Best Performance:

Decision Tree (0.84) and Random Forest (0.81) gives best performance compare to other model.

Lowest Performance:

SVM Linear (0.45) consistently performed poorly**.**

Increasing the number of selected feature from 3 to 5 gave best performance in most cases.