Regression - Advertising Example

Welcome, In our Previous session we Learnt about a Machine Learning Technique known as Classification. Today we learn a New technique known as Regression. Regression is where a Machine Leaning Model predicts a Continous variable i.e Predict temparature, marks, money, sales, profits, age , weight, salaries etc.
Regression is applicable to many areas where continous variable are to be predicted.

NB: In Classification we predicted a Categorical variable/or a Group i.e a Muffin or Cupcake? In this example, you will predict Sales, The predictors will be TV, Radio and Newspaper based on advertising data. Advertising is very important in Business Development and Profit making in todays world of business.

Scenario: A company wanted to create a Machine Learning Model to predict Sales given the expenses spent on Tv, Radio and Newspaper advertising, The dataset accessed below shows amount Spend in TV, Radio and Newspaper(Amount in USD) and corresponding Sales Made in Units.

Regression is a Supervised Learning where we Train and Test the Model.

Step 1: We read the dataset below into a dataframe.

Above we read data into a Dataframe. They are 200 Records each showing TV, Radio, Newspaper expenses and their corresponding Sales in Units

```
import pandas
data = pandas.read_csv('http://modcom.co.ke/datasets/Advertising.csv')
data
```

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	9.7
197	177.0	9.3	6.4	12.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	13.4

200 rows × 4 columns

Above we read data into a Dataframe. They are 200 Records each showing TV, Radio, Newspaper expenses and their corresponding Sales in Units

Step 2: Next, Split the data Specifying the X(feature/Predictors), We use Variable X to hold them. Split the array from 0 - 3, means we only pick Tv, Radio and Newspaper for X and variable Y(target/output) holds only the Sales column.

```
array = data.values
X  = array[:, 0:3]  # TV, radio, newspaper
Y = array[:, 3] # Sales
```

Step 3: Take our Variable X and Y and Split them in 70%(Training Set) and 30%(Testing Set).
test_size = 0.3 indicates the testing is 30%. random_state=42 means we return a shuffled dataset and have a fair split.
X_train and Y_train holds the Training Set. X_test and Y_test holds the Testing Set.

```
from sklearn import *
X_train, X_test , Y_train, Y_test = model_selection.train_test_split(X,Y,
test_size= 0.3, random_state=42)
```

Step 4: Cross Validation
Cross validation involves bringing all models at test them on the Training Dataset, Cross validation gives us an idea of which model is likely to predict well, KFOLD is a technique used to do Cross Validation on Different Models. Same as we did in Classification. Please note the model used are for Regression.

```
# Imports Regression Models
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression

# Create an Empty List and append all models, Give the Models an alias Name
# i.e we called KNeighborsRegressor()) alias KNR etc
models = []
models.append(('D Trees', DecisionTreeRegressor()))
models.append(('KNR', KNeighborsRegressor()))
models.append(('LR', LinearRegression()))
models.append(('Support Machines', SVR(gamma = 'auto')))
models.append(('Random Forest', RandomForestRegressor()))
models.append(('Gradient Boosting', GradientBoostingRegressor()))
models.append(('Support Machines', SVR(gamma = 'auto')))

# Import Cross Validation and KFOLD
from sklearn.model_selection import cross_val_score, KFold
# Create a for loop so that each model is tested in turn
for name, model in models:
     # Here we do a 10 split K-FOLD
     kfold = KFold(n_splits = 10, random_state=42, shuffle=True)

     # We get the results for each Fold
     cv_results = cross_val_score(model, X_train, Y_train, cv = kfold,
     scoring='r2')

     # Get the average of all Folds
     print(name, '  Results:= ', cv_results.mean())
```

```
D Trees   Results:=  0.9035977934001
KNR   Results:=  0.8790725161630053
LR   Results:=  0.8757671101319765
Support Machines   Results:=  -0.1642233881346364
Random Forest   Results:=  0.9662307562718511
Gradient Boosting   Results:=  0.9670361280618144
Support Machines   Results:=  -0.1642233881346364
```

Step 5: From above Cross Validation, we can see that Gradient Boosting Model validates higher with 96.6%, Hence we can Pick it as our training Model


```
from sklearn.ensemble import GradientBoostingRegressor
model = GradientBoostingRegressor()
model.fit(X_train, Y_train)
```

```
▾ GradientBoostingRegressor
GradientBoostingRegressor()
```

Step 6: Testing our model performance using X_test data.

```
predictions = model.predict(X_test)
print("Predicted: ", predictions)
print("Execped: ", Y_test)
```

```
Predicted:  [17.11177509 21.89584916 20.31149838  6.18264522 23.52855497 12.969581
 22.7833106   9.34468943 11.76430183 15.90246377  8.35901421  8.83027211
 12.5139839   2.88190296 10.42048147 12.11370977  4.66835775 16.57238947
 11.31385106 19.16646081 20.32496807 13.27493867 10.6955329  22.6768956
  9.95845348  8.56160215 22.82893407 12.50256299 10.11953957  4.43821117
 11.40986866 11.31385106 22.28843255  8.86464478 15.59768903 20.65114169
 12.39722638 20.19353127 12.44261317  7.27286007 10.79548005 12.64358481
 10.11585805  9.43597287 11.75264503  8.10889839 10.46473459 14.19871042
 10.25025792 12.43543084 14.59689258 12.06376764  6.38757192  4.61386074
  8.8413915  10.88775312 10.22398563 25.32574856  7.03554193 11.78598809]
Execped:  [16.9 22.4 21.4  7.3 24.7 12.6 22.3  8.4 11.5 14.9  9.5  8.7 11.9  5.3
 10.3 11.7  5.5 16.6 11.3 18.9 19.7 12.5 10.9 22.2  9.3  8.1 21.7 13.4
 10.6  5.7 10.6 11.3 23.7  8.7 16.1 20.7 11.6 20.8 11.9  6.9 11.  12.8
 10.1  9.7 11.6  7.6 10.5 14.6 10.4 12.  14.6 11.7  7.2  6.6  9.4 11.
 10.9 25.4  7.6 11.7]
```

Step 7: Check accuracy with R Squared. In Regression R Squared is a measure of how well the model performed on test data out of 100. Below the model scored 97% hence very accurate.

```
from sklearn.metrics import r2_score
print("Accuracy ", r2_score(predictions,  Y_test))
```

```
Accuracy  0.9815180207795777
```

Step 8: Prediction

Assume the company have a budget 400 USD for advertising, Lets see if the model can help us distribute the 400 USD and get the optimum/best Sales in Units, Below we provide 200 USD for TV, 100 for Radio and 100 for Newspaper, The outcome is 22 Units Sales

```
expense = [[200, 100, 100]]
outcome = model.predict(expense)
print("You will sell ", outcome, 'Units')
```

```
You will sell  [22.07365409] Units
```

Change the amount distribution such that TV has the highest expense, The Unit sales change to 26 Units below, Hence we can learn that spending more on TV, a bit on 100 and now Newspaper could give you higher Unit Sales.

```
expense = [[300, 100, 0]]
outcome = model.predict(expense)
print("You will sell ", outcome, 'Units')
```

```
You will sell  [26.76592205] Units
```