PREDICTIVE ANALYTICS

36_Ashana Mehta

1. Problem Statement

(a) To predict whether a firm will manufacture or not.

(b) To determine cash compensation for employees of a firm.

2. Data Description

```
data = pd.read_excel("C:\\Users\\ASUS\\Downloads\\Logistic Regression .xlsx")
data.head()
```

	Cash Compensation	Sales	No.of Employees	Capital Investment	Manufacturing
0	212	32.0	248	10.5	1
1	226	27.2	156	3.8	0
2	237	49.5	348	14.6	1
3	239	34.0	196	5.0	0
4	242	52.8	371	15.9	1

The database consists of 22 rows and 5 columns. The features are:

- Cash Compensation
- Sales
- No. of Employees
- Capital Investment
- Manufacturing

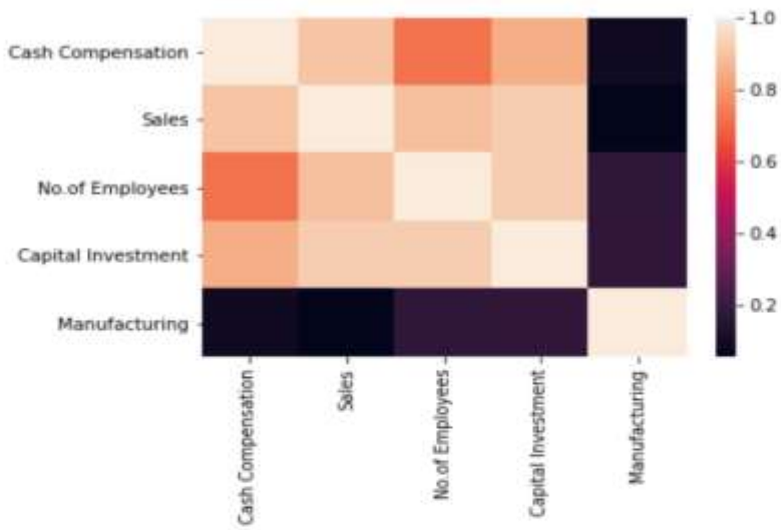
For problem (a), 'Manufacturing' is considered as the dependent variable.

For problem (b), 'Cash Compensation' is considered as the dependent variable and 'Manufacturing' is not used because of least correlation shown below.

	Cash Compensation	Sales	No.of Employees
Cash Compensation	1.000000	0.903011	0.722292
Sales	0.903011	1.000000	0.891040
No.of Employees	0.722292	0.891040	1.000000
Capital Investment	0.850533	0.925308	0.923409
Manufacturing	0.090179	0.053149	0.171090

	Capital Investment	Manufacturing
Cash Compensation	0.850533	0.090179
Sales	0.925308	0.053149
No.of Employees	0.923409	0.171090
Capital Investment	1.000000	0.172671
Manufacturing	0.172671	1.000000

<AxesSubplot:>



For the dataset, there is high correlation among all the variables except the categorical variable ‘Manufacturing’.

3. Train Test Split

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(16, 4)
(6, 4)
```

The data was split using the 'train_test_split' function. 'X_train' has 16 rows while 'X_test' has 6 rows.

4. Analytical Techniques

4.1 Logistic Regression

```
##Logistic Regression
X=data.drop(["Manufacturing"],axis=1).copy() # Define dependent and independent variable
Y=data["Manufacturing"]
lor = LogisticRegression()
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random_state=0) # Split data into training and test set (75:25
train=lor.fit(X_train,Y_train)
pred=lor.predict(X_test)
```

```
C:\Users\ASUS\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver will be changed t
o 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
print("The intercept is : ",lor.intercept_)
print("The coefficients of other variables are : ",lor.coef_)
```

```
The intercept is :  [0.00616795]
The coefficients of other variables are :  [[-0.00630993 -0.72888411  0.1132008   0.11502436]]
```

The logistic regression model is build using the 'scikit-learn' library.

The intercept of the model is **0.00616795** and the coefficients of the independent variables are:

- Cash Compensation: **-0.0063**
- Sales: **-0.7288**
- No. of Employees: **0.1132**
- Capital Investment: **0.115**

Confusion Matrix

```
# Define the confusion matrix for the model on testing data
cm = confusion_matrix(Y_test, pred)
print ("The confusion matrix is : \n",cm)
```

```
The confusion matrix is :
 [[1 0]
 [1 4]]
```

```
#The classification table for testing data
print(classification_report(Y_test, pred))
```

```
              precision    recall  f1-score   support

           0       0.50      1.00      0.67         1
           1       1.00      0.80      0.89         5

    accuracy                           0.83         6
   macro avg       0.75      0.90      0.78         6
weighted avg       0.92      0.83      0.85         6
```

```
print ("Accuracy for test data : ", accuracy_score(Y_test,pred)) #accuracy score
print ("Precision for test data : ", precision_score(Y_test,pred))
print ("Recall_score for test data : ", recall_score(Y_test,pred))
print ("F1 score for test data : ", f1_score(Y_test,pred))
print ("Accuracy : ", accuracy_score(Y_train,tr1_pred)) #accuracy score #comes to
lor.predict_proba(X_train) #Predicting probabilities of the independent variables
```

```
Accuracy for test data :  0.8333333333333334
Precision for test data :  1.0
Recall_score for test data :  0.8
F1 score for test data :  0.888888888888889
Accuracy :  1.0
```

From the confusion matrix we have, TN = 1, FP = 0, FN = 1, TP = 4.

- Accuracy = 0.833
- Precision = 1
- Recall = 0.8
- F1 score = 0.889

The last mentioned accuracy score is derived when training data is put for accuracy test.

4.2 Multiple Regression

```
model=sm.OLS(Y_train,X_train)
results=model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:      Cash Compensation   R-squared:                       0.984
Model:                            OLS   Adj. R-squared:                  0.979
Method:                 Least Squares   F-statistic:                     239.1
Date:                Tue, 08 Sep 2020   Prob (F-statistic):           5.78e-11
Time:                        22:54:22   Log-Likelihood:                -65.278
No. Observations:                  16   AIC:                             138.6
Df Residuals:                      12   BIC:                             141.6
Df Model:                           3
Covariance Type:            nonrobust
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                216.6434      6.683     32.419      0.000     202.083     231.204
Sales                  1.6371      0.318      5.145      0.000       0.944       2.330
No.of Employees       -0.0199      0.037     -0.536      0.602      -0.101       0.061
Capital Investment    -3.2732      0.501     -6.530      0.000      -4.365      -2.181
==============================================================================
Omnibus:                        5.126   Durbin-Watson:                   2.315
Prob(Omnibus):                  0.077   Jarque-Bera (JB):                2.436
Skew:                           0.770   Prob(JB):                        0.296
Kurtosis:                       4.133   Cond. No.                     3.43e+03
```

The model is first built using the 'statsmodels' library.

The R-squared is 0.984 while the adjusted R-squared is 0.979.

The p-value of the overall model is very close to 0.

The multiple regression equation is

$$\begin{aligned} Cash\ Compensation \\ &= 216.6434 + 1.6371 \times Sales - 0.0199 \times No.\ of\ Employees \\ &- 3.2732 \times Capital\ Investment \end{aligned}$$

The intercept, 'Sales' and 'Capital Investment' are significant in the model while 'No. of Employees' has a very high p-value of 0.602.

```
from sklearn.linear_model import LinearRegression

X_train = X_train[['Sales', 'No.of Employees', 'Capital Investment']]

model2 = LinearRegression()
model2.fit(X_train, y_train)
y_pred = model2.predict(X_test)
y_pred
```

```
array([-38.15380861, 286.86700394, 350.11176671, 329.71918618,
       245.62463571,  36.46240292])
```

The values for the test data is predicted as shown.

5. Conclusion

Both the models work well for the dataset. The logistic regression model gives high values accuracy, recall, precision and F1 score and the multiple regression model gives a very high R-square.

Thus, manufacturing and cash compensation can be predicted with minimal error using the model provided we remove the "manufacturing" column while using multiple linear regression.