A logistic model to predict if preliminary year students will get into college

The Issue:

In this project we have the data of students who are about to complete preliminary year. By building a logistic model we have to predict the success or failure of preliminary student to get admission into college.

There are total of 33 factors contributing to the final result whether a particular student getting admission into college will be success or failure. Some factors might be having high significance and some might not contribute much in building the logistic model. Our aim is to build a logistic model and find out what all factors will be significant for the end result.

Findings:

From the student data containing total of 33 factors, for which we made a logistic model to predict success or failure of students who are doing a preliminary year to see if they will be admitted to college. We can calculate accuracy, precision, recall value and F1-score values to know how well our logistic model performed.

From the logistic model we build, we found out that the accuracy of the logistic model is "0.86363", precision of the logistic model is "0.8462", recall of the logistic model is "0.9167" and the F1-score of the logistic model is "0.88".

Discussion:

From the model we built and the results we obtained there are few discussions we can put on table. Firstly, from the accuracy we can say that we can say that we have 86.36% correct prediction over total predictions, which is a good percentage. Next, from precision we can say that how good is our model when the prediction is positive. Recall measures how good our model at correctly predicting positive values. F1-score is a machine learning evaluation metric that measure's a model's accuracy. Considering all the values of accuracy, precision, recall and F1-score we can say that the logistic model which we built is a good model.

Appendix A: Method

We perform logistic regression on a dataset to predict success or failure of students who are doing a preliminary year to see if they will be admitted to college.

First, we import dataset using pd.read_excel() function. It reads the data and we can get the information about our dataset using data.info() function.

Next we start cleaning the given data, that is dropping the unnecessary columns, filling the empty or not applicable values with mode of the respective factor and converting any object values into int or float values using label encoder. We push all this cleaned data into a new data variable.

In the given scenario, the dataset is first divided into two parts, one for training and the other for testing, using an 80:20 ratio. This means that 80% of the data will be used for training and 20% for testing. After the data is divided, the training data is used to train the logistic regression model. Once the model is trained, it is tested on the test data to evaluate its performance. The logistic regression model predicts the class of the test data and the predicted values are compared with the actual values. The accuracy of the model is calculated by comparing the predicted values with the actual values.

From the above created logistic model, we get the desired results, that is accuracy, precision, recall and F1-score values.

Then we try to get the coefficients of various vectors of the new data and we check the positively correlated factors and negatively correlated factors.

Appendix B: Results

From the logistic model we built the following are the results,

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print('Accuracy:', accuracy)
print("Precision: ", precision)
print("Recall: ", recall)
print("F1-score: ", f1)
```

```
Accuracy: 0.8636363636363636
Precision:  0.8461538461538461
Recall:  0.9166666666666666
F1-score:  0.8799999999999999
```

And coefficients are shown as below,

	feature	coefficient
21	Completed Community Service Requirement? (1=ye...	1.216846
24	Number of Workshops Attended	0.922985
29	Completed Connect? (1=yes, 0=no)	0.918969
2	Federal Ethnic Group	0.915511
23	Number of Peer Mentor Meetings Attended	0.588715
5	Attended Orientation? (1=yes, 0=no)	0.412935
20	Completed Campus Event Requirement? (1=yes, 0=no)	0.401207
25	F17 GPA	0.299432
9	Completed Summer Bridge? (2=completed all, 1=c...	0.275527
6	Attended Experience Day? (1=yes, 0=no)	0.247443
28	Number of Credits Earned	0.216230
8	Athlete? (1=yes, 0=no)	0.121882
14	Receptivity to Academic Assistance (percentile...	0.072988
22	Number of Faculty Advisor Meetings Attended	0.059056
10	Dropout Proneness (percentile score before sta...	0.053809
16	Receptivity to Social Engagement (percentile s...	0.050450
17	Receptivity to Career Guidance ((percentile sc...	0.038631
15	Receptivity to Personal Counseling (percentile...	0.012981
1	SAT Score	0.001108
18	Receptivity to Financial Guidance (percentile ...	-0.003163
12	Educational Stress (percentile score before st...	-0.007368
19	Desire to Transfer (percentile score before st...	-0.009066
27	CUM GPA	-0.040823
4	Pell Grant Eligible? (1=yes, 0=no)	-0.044285
11	Predicted Academic Difficulty (percentile scor...	-0.064198
13	Receptivity to Institutional Help (percentile ...	-0.122403
26	S18 GPA	-0.225562
3	Gender	-0.248215
7	Resident/Commuter (1=resident, 0=commuter)	-0.278693
0	High School GPA	-0.307349

Appendix C: Code

Code is written in python language with various packages like pandas, numpy and sklearn.

-

```
import numpy as np
import pandas as pd
from sklearn import preprocessing
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

```
#importing data
data = pd.read_excel('/kaggle/input/studx1/Preliminary college year (1).xlsx')
data.head()
data.info()
```

```
#cleaning data
data1 = data.fillna(data.median(numeric_only=True))
data1.info()
data1.drop('Reason not Retained',axis =1,inplace = True)
data1.drop('Reason for not Completing Connect',axis = 1,inplace = True)
```

```
data1['Completed Connect? (1=yes, 0=no)'].value_counts()
data2 = data1.loc[data1['Completed Connect? (1=yes, 0=no)'] != '0, contract for fall']
data2['Completed Connect? (1=yes, 0=no)'].value_counts()
data2.info()
```

```
data2['Federal Ethnic Group'] = data2['Federal Ethnic Group'].fillna(data2['Federal Ethnic Group'].mode()[0])
data2['Gender'] = data2['Gender'].fillna(data2['Gender'].mode()[0])
data2['Completed Connect? (1=yes, 0=no)'] = data2['Completed Connect? (1=yes, 0=no)'].fillna(data2['Completed Connect? (1=yes, =no)'].mode()
data2.info()
```

```
data2['Completed Connect? (1=yes, 0=no)'] = data2['Completed Connect? (1=yes, 0=no)'].astype(str).astype(float)
data2['Federal Ethnic Group'] = data2['Federal Ethnic Group'].astype(str)
data2['Gender'] = data2['Gender'].astype(str)
data2.info()
```

```
# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()
data3 = data2
data3['Federal Ethnic Group'] = label_encoder.fit_transform(data3['Federal Ethnic Group'])
data3['Gender'] = label_encoder.fit_transform(data3['Gender'])
data3.head()
data3.info()
```

```
#splitting data
X_train, X_test, y_train, y_test = train_test_split(data3.drop('Retained F17-F18? (1=yes, 0=no)', axis=1), data3['Retained F17-F18? (1=yes, 
```

```
#logistic model
model = LogisticRegression(max_iter=8000)
fit = model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

```
#results of logestic model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
print('Accuracy:', accuracy)
print("Precision: ", precision)
print("Recall: ", recall)
print("F1-score: ", f1)
```

```
#coefficients of various vectors in descending order
coef = fit.coef_[0]
features = X.columns
sorted_indices = np.argsort(coef)
coefficients = pd.DataFrame({"feature":X_train.columns,"coefficient":fit.coef_[0]})
coefficients = coefficients.sort_values(by = "coefficient",ascending = False)
print(coefficients)
```