EMPLOYABILITY PREDICTOR USING MACHINE LEARNING

-PADMASHRI B V

1.PROBLEM STATEMENT

In today's job market, students and graduates face lots of struggle to find a job in every industry like in the field of information technology,banking sector,automobile sector and almost all the industries. The problem addressed by the Employability Predictor app is the lack of personalized guidance for students who want to improve their employability. Many students graduate with a limited understanding of their strengths and weaknesses in the job market today.The application aims to bridge this gap by providing tailored insights and recommendations based on the user's skills and interest updated in the profile.

2. MARKET/CUSTOMER/BUSINESS NEED ASSESSMENT

- Today the job market is highly competitive, and students often lack guidance on how to improve their employability
- There is a need of personalized recommendations to make informed decisions about their education and skill development
- The students and recent graduates needs career guidance ,skill improvement,and job market insights
- The businesses can address the gap in the market and provide a valuable service while potentially generating revenue through premium features and partnerships with Top HR's.

3.TARGET SPECIFICATIONS AND CHARACTERIZATION (Customer Characteristics)

- **The primary objective of the Employability Predictor App is to develop machine learning models to calculate personalized employability scores for users**
- **And create a user-friendly interface for profile creation, employability score prediction, and personalized recommendations**
- **Our targets are students and recent graduates(age 18-30)from various educational backgrounds**
- **The app provide insights into job market trends and career pathways**
- **Ensure data privacy and security in compliance with regulations**
- **Continuously improve the app through user feedback**

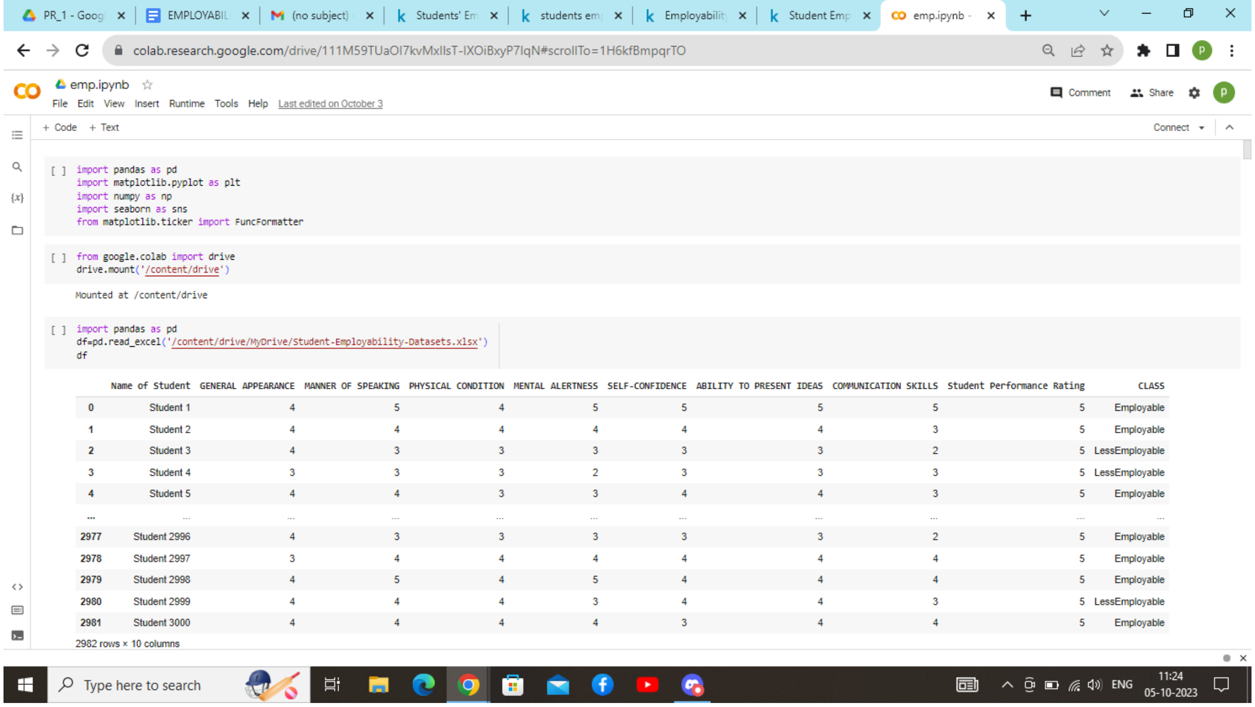
4.EXTERNAL SEARCH(INFORMATION SOURCES/REFERENCES)

The usage of Student's employability dataset-philippines as base data for this project

Link to this dataset:https://www.kaggle.com/datasets/anashamoutni/students-employability-dataset

The dataset is found in kaggle. The dataset consists of Mock job Interview Results of 2982 observations. Dataset was collected from different university agencies in the Philippines. The dataset that was collected is compliant with the Data Privacy Act of the Philippines.

DATASET



```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from matplotlib.ticker import FuncFormatter
```

```
from google.colab import drive
drive.mount('/content/drive')
```

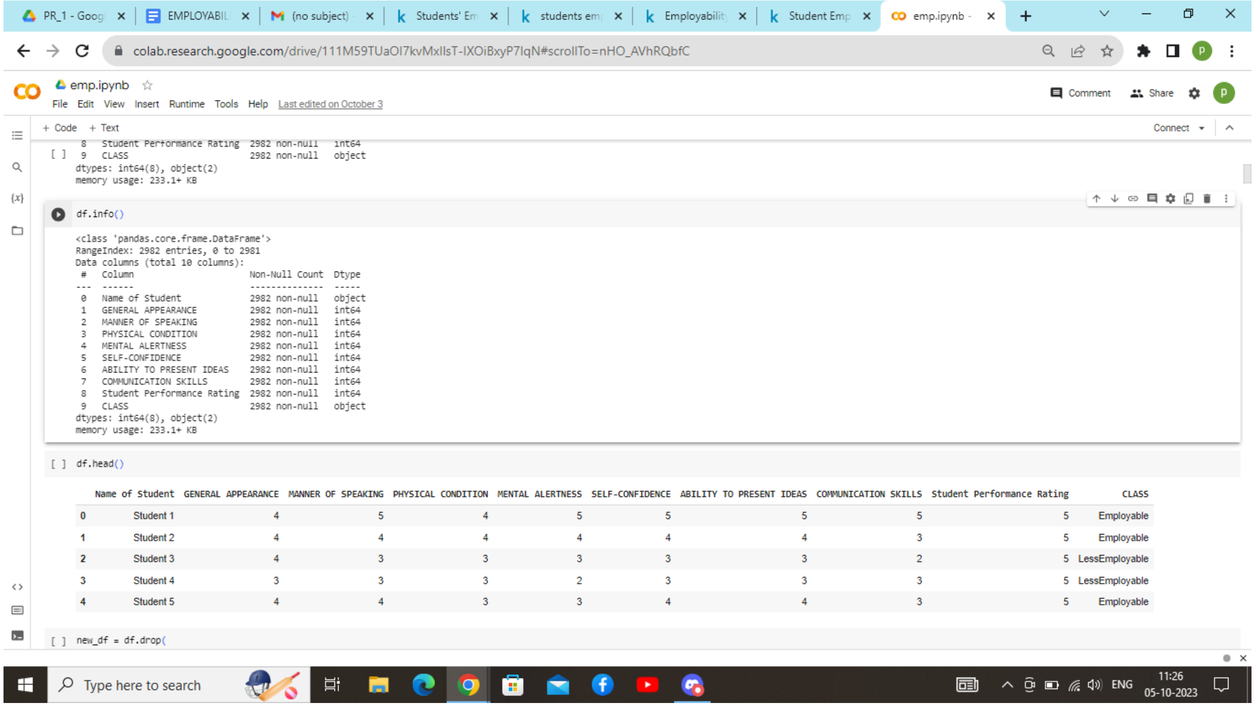
```
Mounted at /content/drive
```

```
import pandas as pd
df=pd.read_excel('/content/drive/MyDrive/Student-Employability-Datasets.xlsx')
df
```

	Name of Student	GENERAL APPEARANCE	MANNER OF SPEAKING	PHYSICAL CONDITION	MENTAL ALERTNESS	SELF-CONFIDENCE	ABILITY TO PRESENT IDEAS	COMMUNICATION SKILLS	Student Performance Rating	CLASS
0	Student 1	4	5	4	5	5	5	5	5	Employable
1	Student 2	4	4	4	4	4	4	3	5	Employable
2	Student 3	4	3	3	3	3	3	2	5	LessEmployable
3	Student 4	3	3	3	2	3	3	3	5	LessEmployable
4	Student 5	4	4	3	3	4	4	3	5	Employable
...	...	...	...	...	...	...	...	...	...	...
2977	Student 2996	4	3	3	3	3	3	2	5	Employable
2978	Student 2997	3	4	4	4	4	4	4	5	Employable
2979	Student 2998	4	5	4	5	4	4	4	5	Employable
2980	Student 2999	4	4	4	3	4	4	3	5	LessEmployable
2981	Student 3000	4	4	4	4	3	4	4	5	Employable

2982 rows × 10 columns

INFORMATION OF THE DATASET



```
 8   Student Performance Rating  2982 non-null   int64
 9   CLASS                       2982 non-null   object
dtypes: int64(8), object(2)
memory usage: 233.1+ KB
```

```
df.info()
```

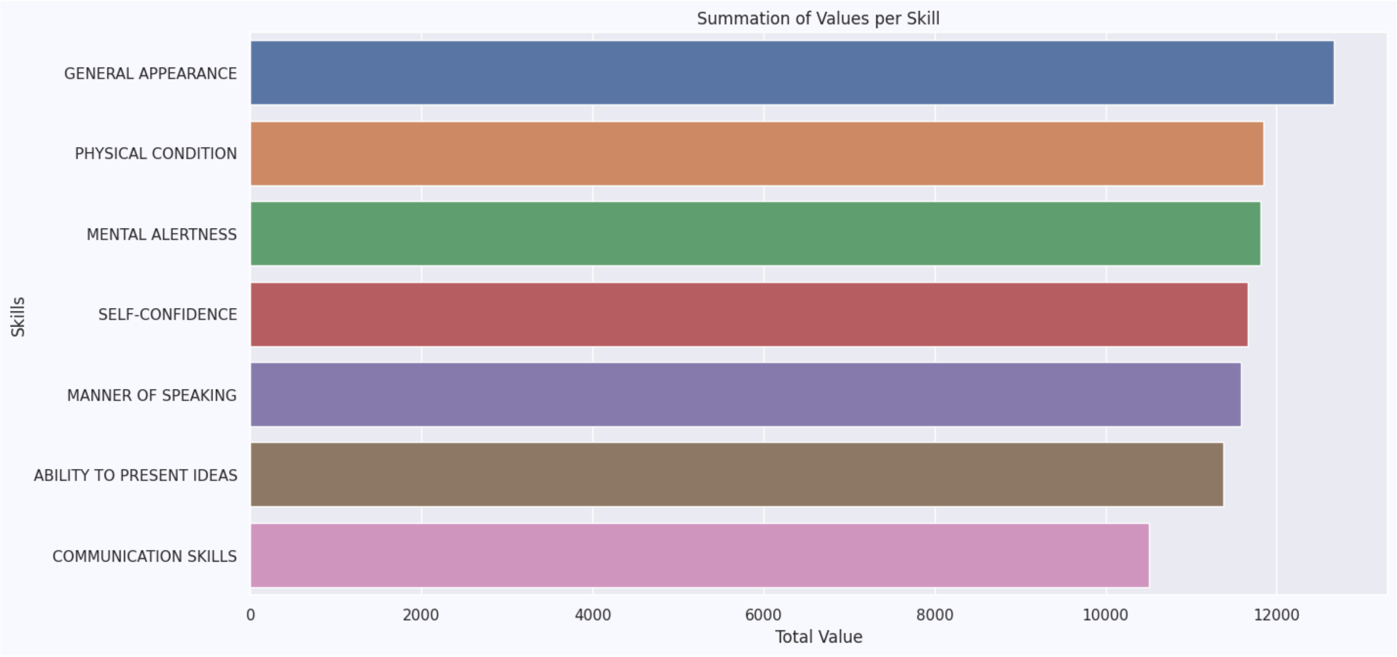
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2982 entries, 0 to 2981
Data columns (total 10 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Name of Student             2982 non-null   object
 1   GENERAL APPEARANCE          2982 non-null   int64
 2   MANNER OF SPEAKING          2982 non-null   int64
 3   PHYSICAL CONDITION          2982 non-null   int64
 4   MENTAL ALERTNESS            2982 non-null   int64
 5   SELF-CONFIDENCE             2982 non-null   int64
 6   ABILITY TO PRESENT IDEAS    2982 non-null   int64
 7   COMMUNICATION SKILLS        2982 non-null   int64
 8   Student Performance Rating  2982 non-null   int64
 9   CLASS                       2982 non-null   object
dtypes: int64(8), object(2)
memory usage: 233.1+ KB
```

```
df.head()
```

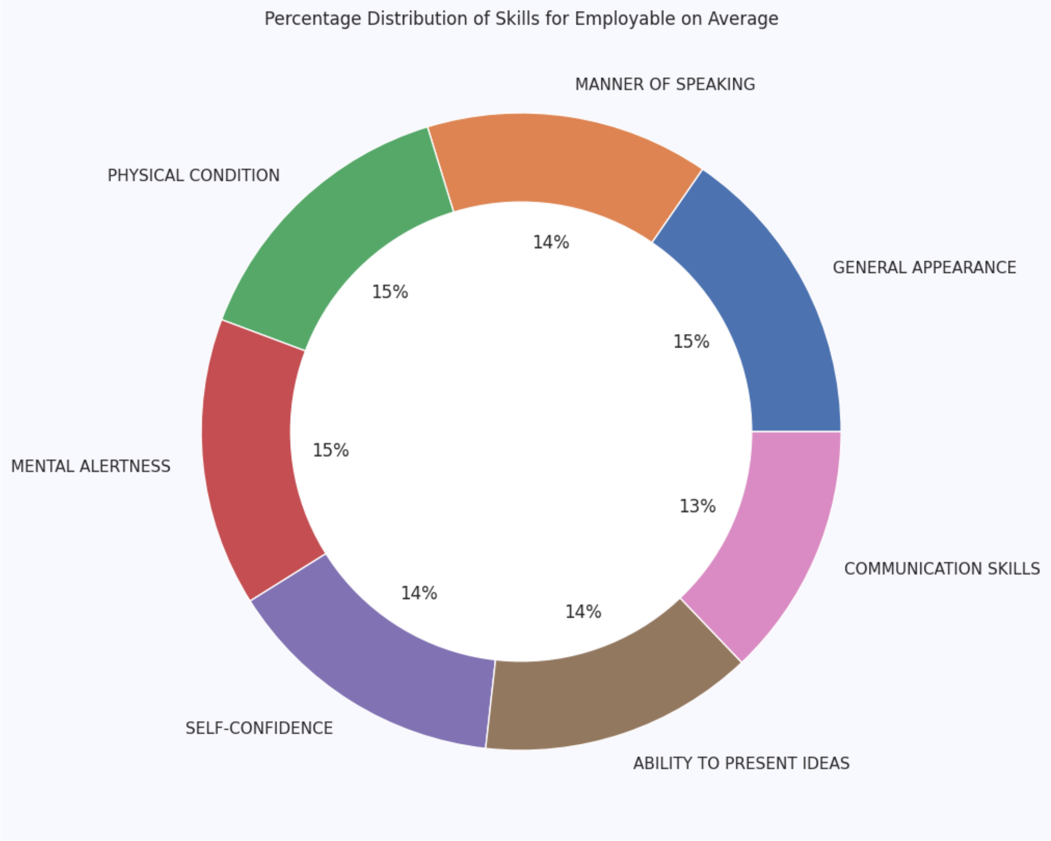
	Name of Student	GENERAL APPEARANCE	MANNER OF SPEAKING	PHYSICAL CONDITION	MENTAL ALERTNESS	SELF-CONFIDENCE	ABILITY TO PRESENT IDEAS	COMMUNICATION SKILLS	Student Performance Rating	CLASS
0	Student 1	4	5	4	5	5	5	5	5	Employable
1	Student 2	4	4	4	4	4	4	3	5	Employable
2	Student 3	4	3	3	3	3	3	2	5	LessEmployable
3	Student 4	3	3	3	2	3	3	3	5	LessEmployable
4	Student 5	4	4	3	3	4	4	3	5	Employable

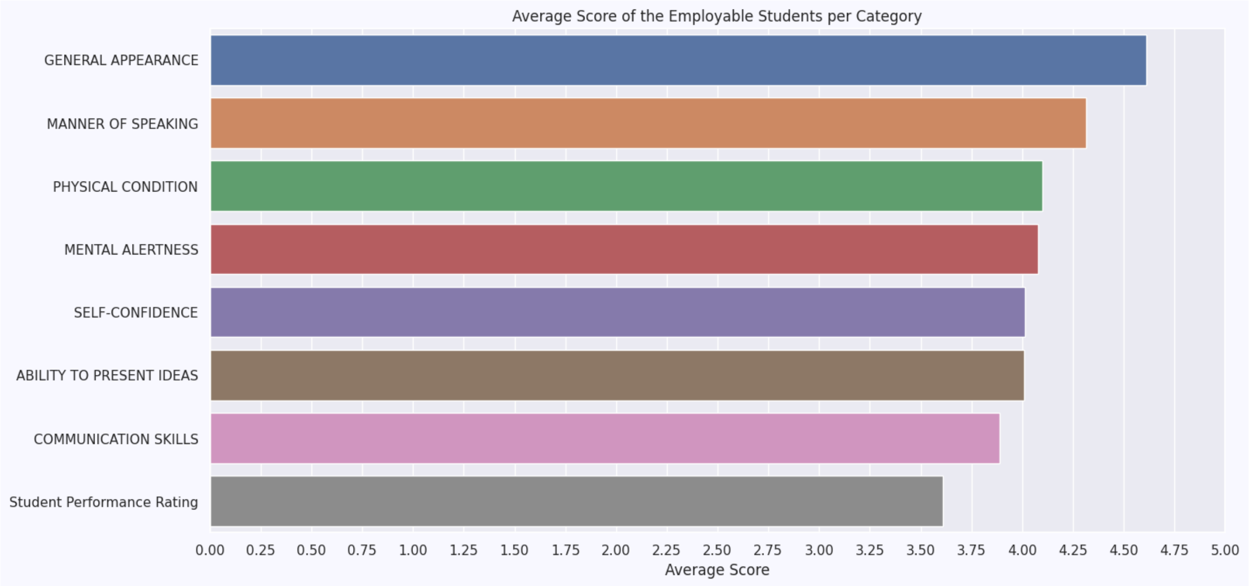
```
new_df = df.drop(
```

5.BENCHMARKING

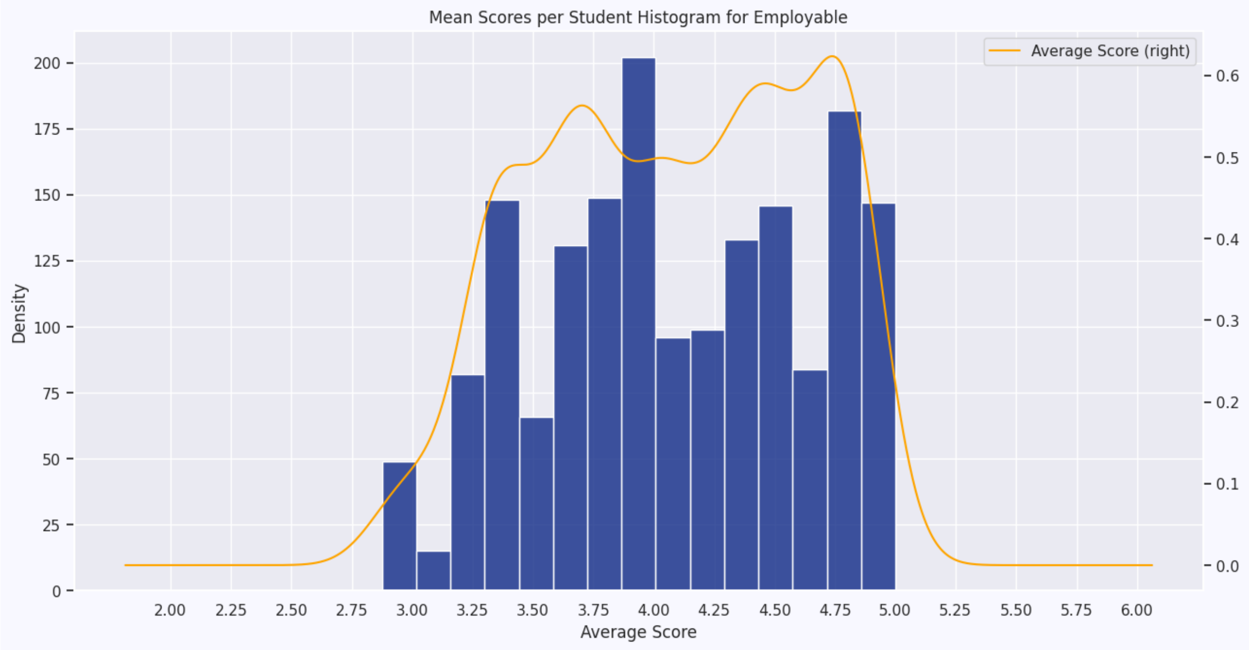


students are lacking in skills but best in general appearence

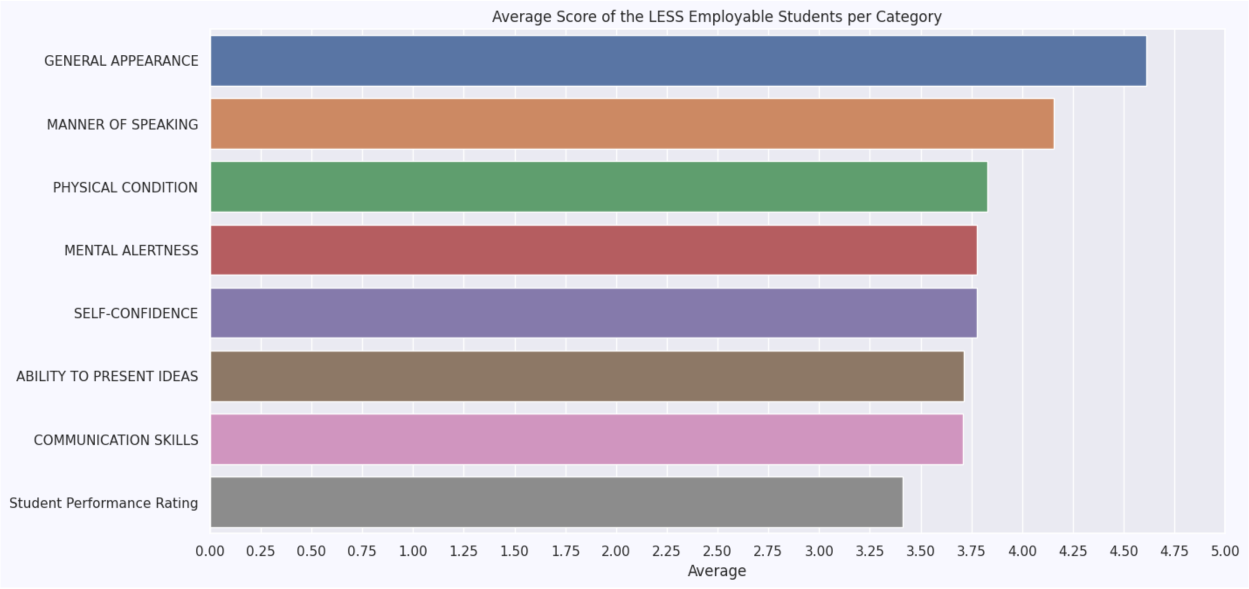




We can see that the mean scores for employable students are capable of reaching a mark of 4 and above for most of the skills

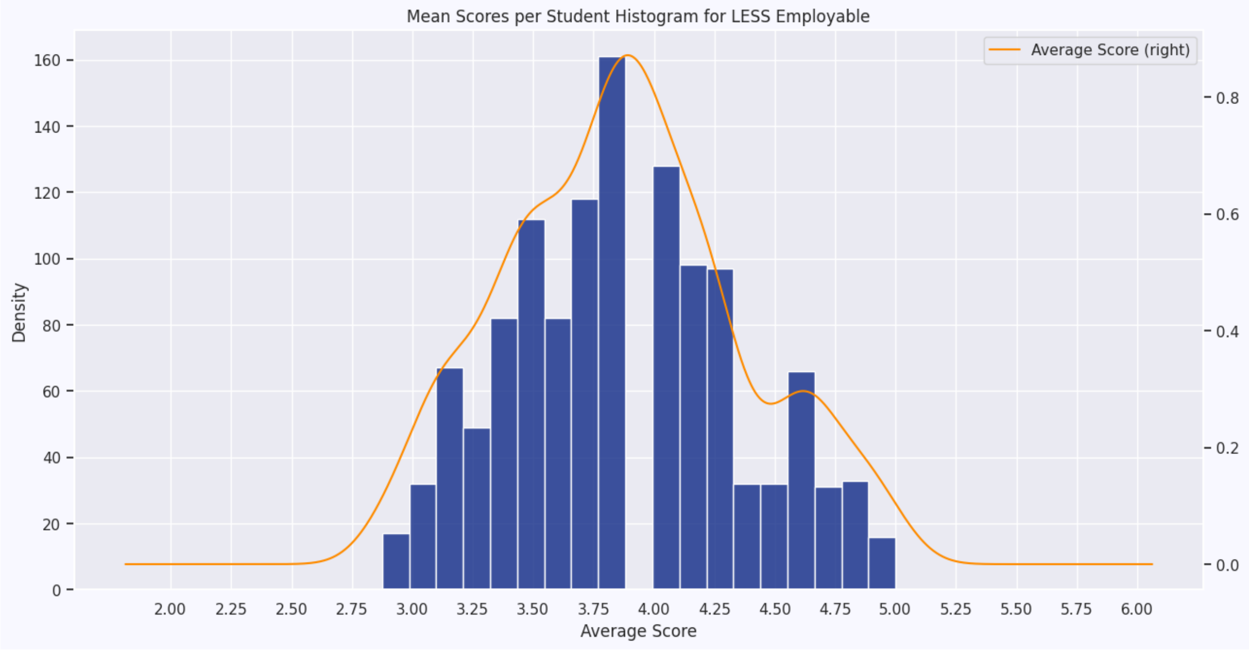


The Histogram shows that the minimum scores for the Employable can be at least 3.0

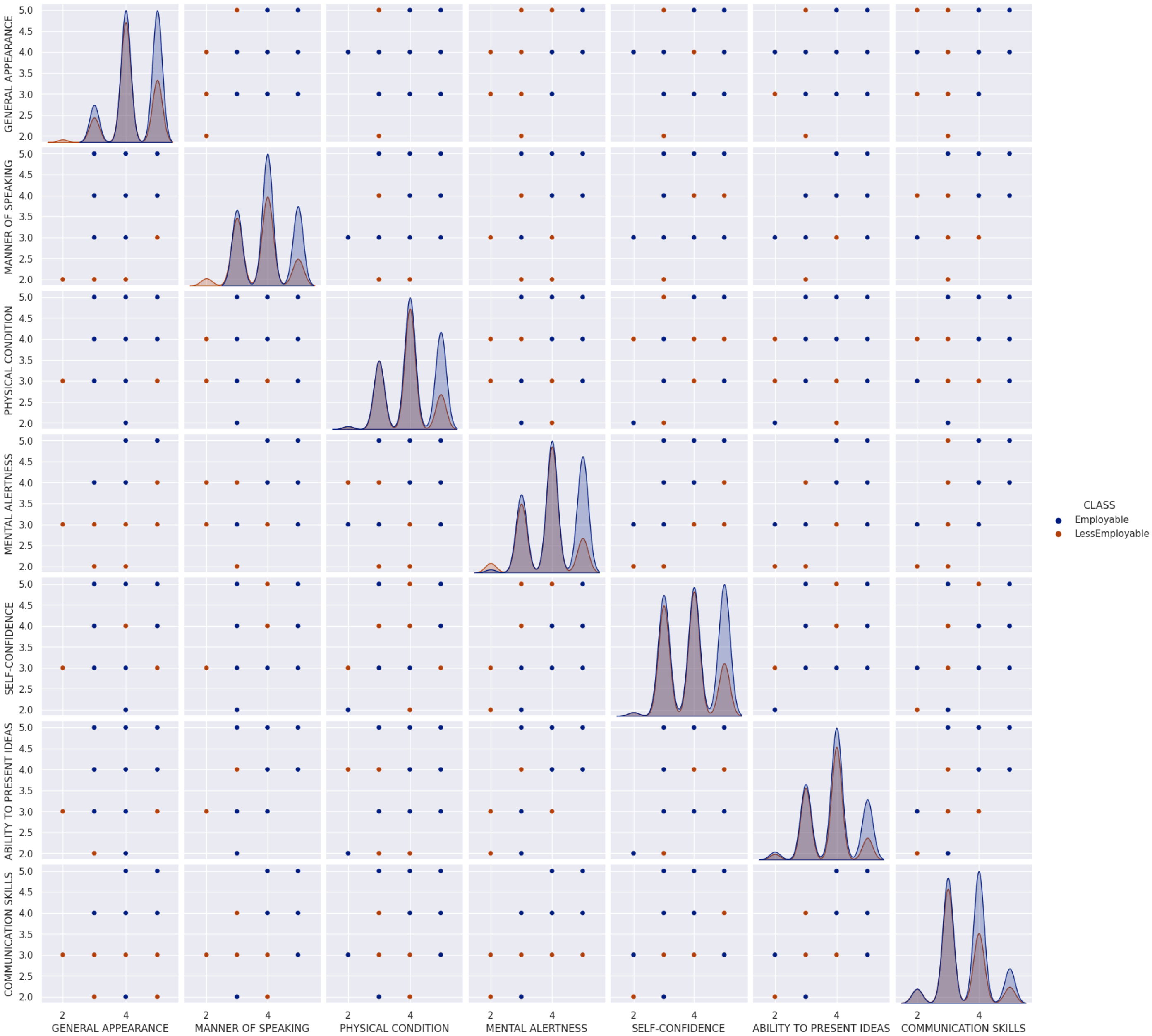


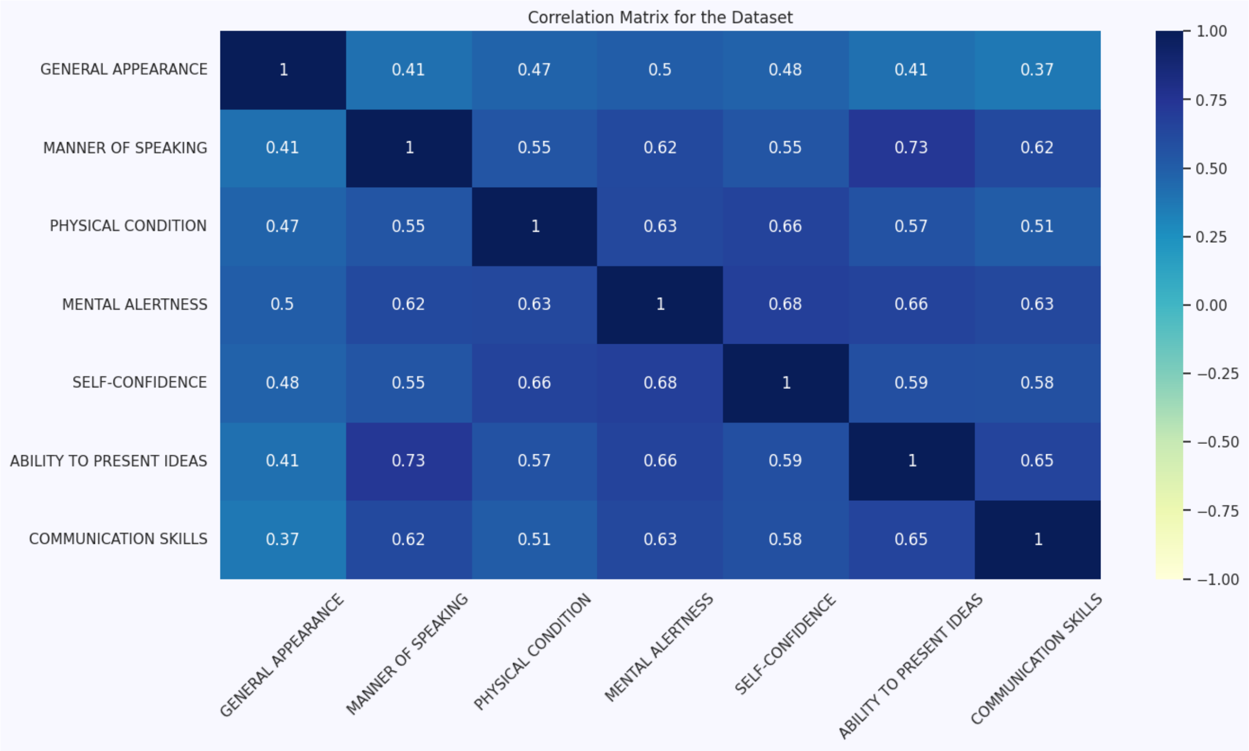
Comparing this to the Employable most of the Categories in Less Employable failed to reach an average score of 4, except for general appearance. Whereas in Employable, 5 categories were able to reach an average score of 4.

In both charts we can see that Students are lacking of Communication skills, but is excelling at General Appearance

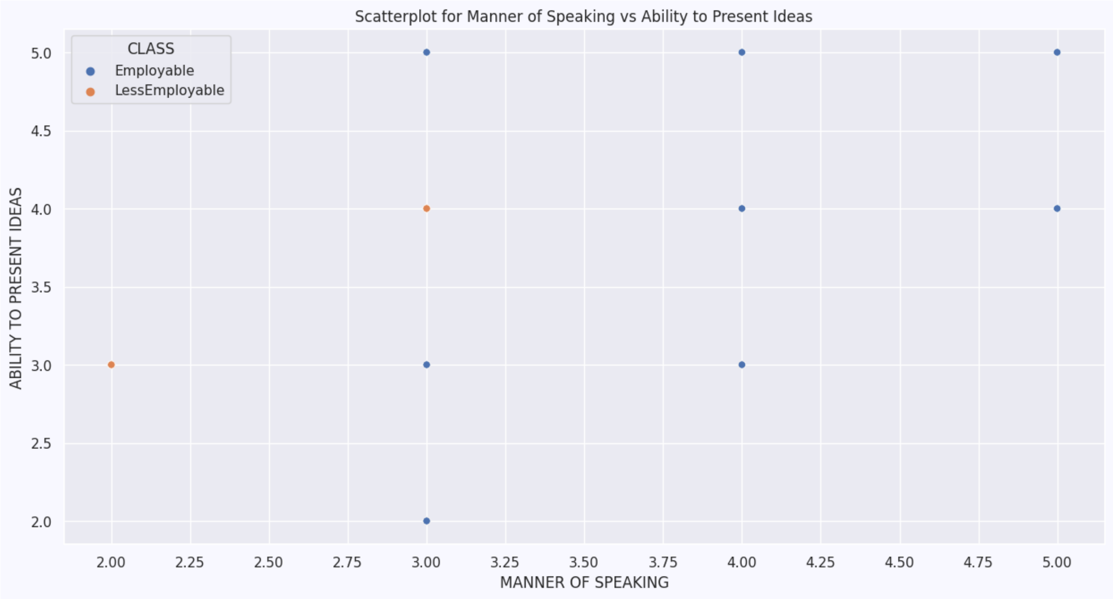


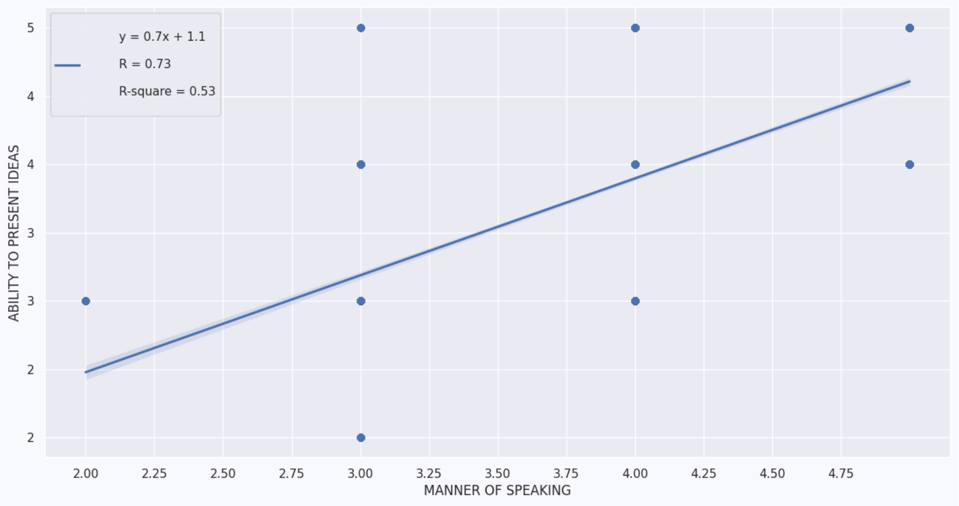
Surprisingly, Less Employable Students can also reach an average score of 4 and higher. This shows that it is possible to be Less Employable even if you can reach an average score higher than 4.


GENERAL APPEARANCE
MANNER OF SPEAKING
PHYSICAL CONDITION
MENTAL ALERTNESS
SELF-CONFIDENCE
ABILITY TO PRESENT IDEAS
COMMUNICATION SKILLS
5.0
4.5
4.0
3.5
3.0
2.5
2.0
2
4
CLASS
Employable
LessEmployable



We can see that there is a strong correlation between Manner of Speaking and The Ability to present ideas with an R of 0.73 and R-square of 0.53





6.APPLICABLE PATENTS

- **Conduct a patent search to ensure there are no infringements in the proposed machine learning models or algorithms**

- **Explore licensing options for any required third-party technologies**

7.APPLICABLE REGULATIONS

- **Ensuring compliance with data privacy regulations like GDPR,CCPA while handling the data**

- **Adhere to guidelines for personalized recommendations**

8.APPLICABLE CONSTRAINTS

- **Limited budget for app development and marketing.**

- **Need for a multidisciplinary team with expertise in machine learning ,app development, and UI/UX design**

9.BUSINESS MODEL(MONETIZATION IDEA)

- **FREE ASSESSMENT**: offering basic employability assessment and recommendations for free and charge for premium features like in-depth career counseling and personalized coaching

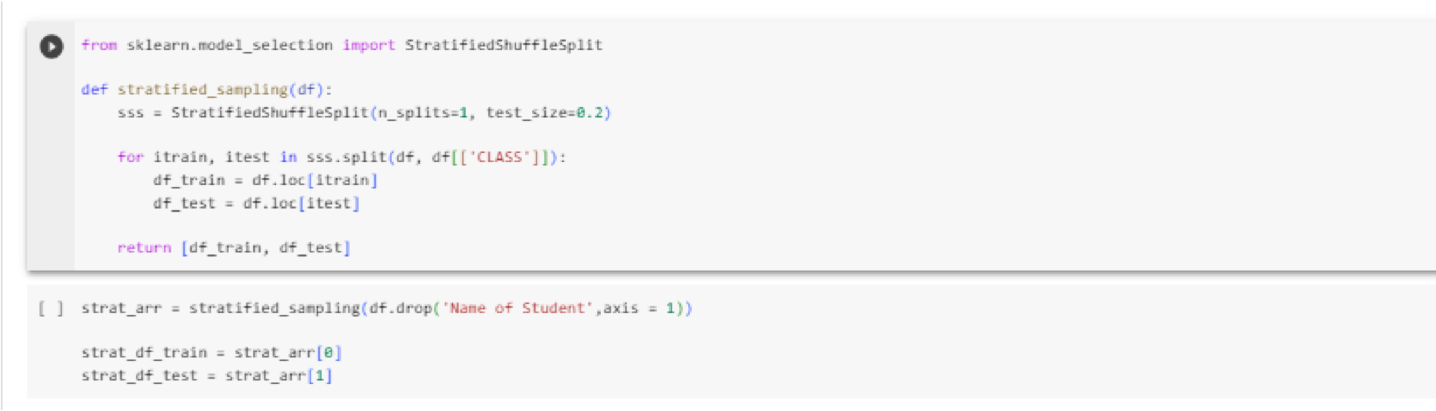
- **SUBSCRIPTION MODEL**: charging a monthly or annual subscription fee for access to advanced features with fare charges compared to others

- **PARTNERSHIP MODEL**: Collaborate with educational institutions for bulk licensing and offer student access as part of their career services

10.CONCEPT GENERATION

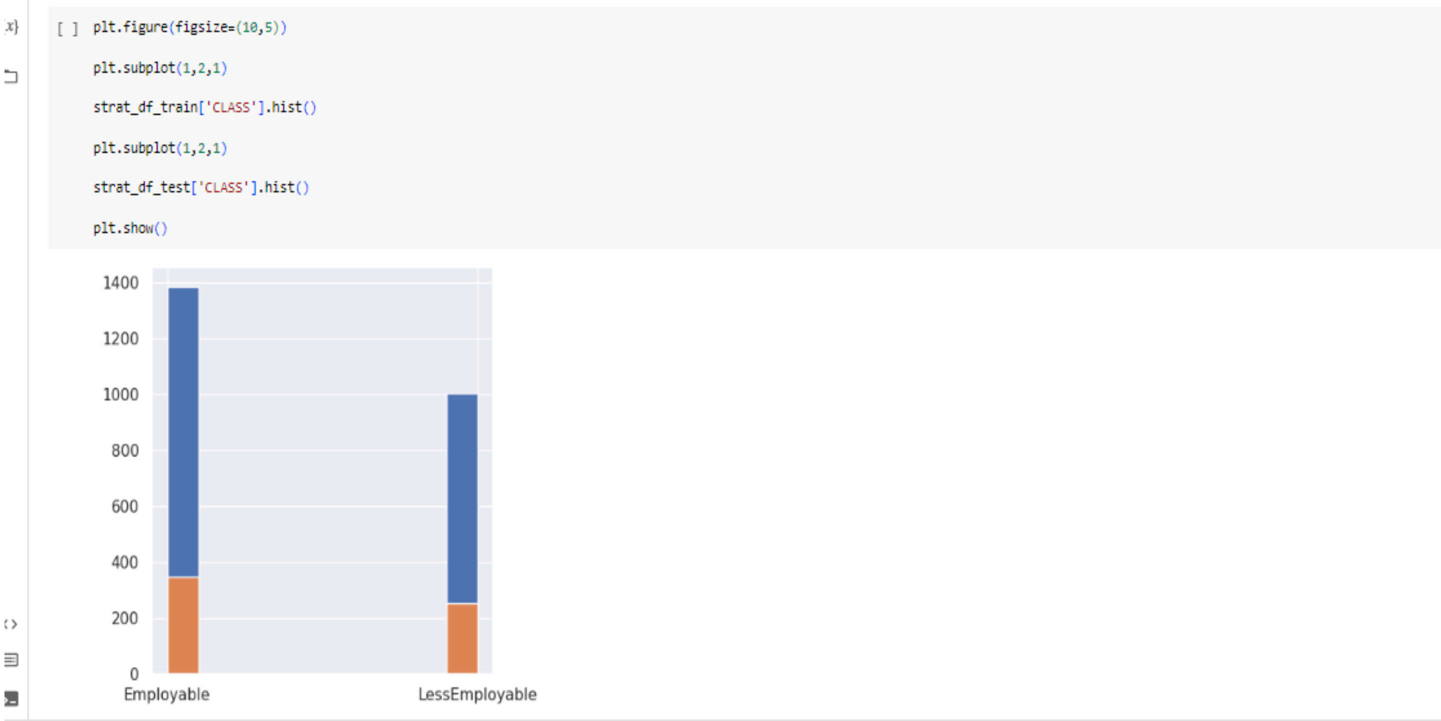
- The idea comes from the need for personalized career guidance in a competitive job market.Inspired by the success of machine learning -driven recommendation systems in various domains. The product requires building a machine learning model from scratch.

  1.import dataset in python environment(google colab)
  2.clean the dataset and analyze the dataset
  3.find correlation and interpret the data
  4.Start building the model, we choose to select two machine learning models in this product: Decision tree classifier and K-nearest classifier.
  5.After building the we will select the best model based on high accuracy

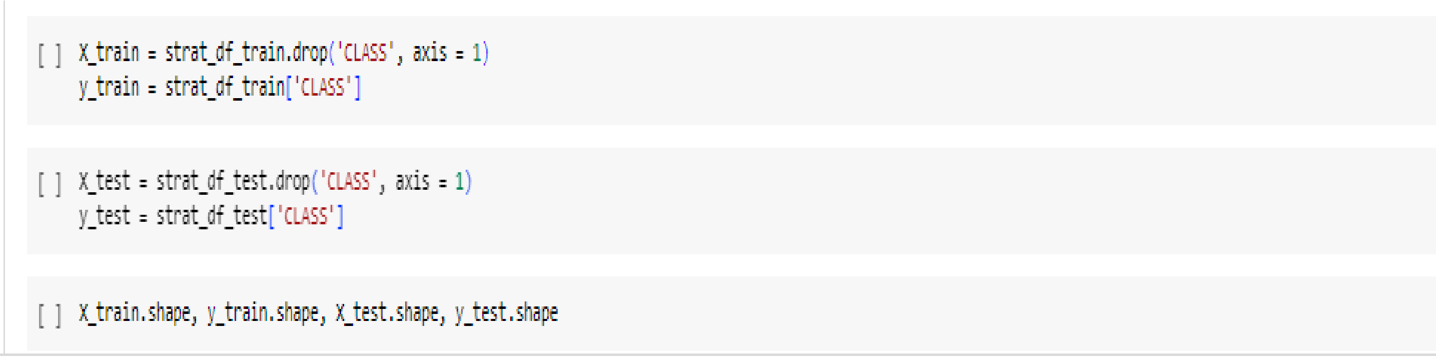


```
from sklearn.model_selection import StratifiedShuffleSplit

def stratified_sampling(df):
    sss = StratifiedShuffleSplit(n_splits=1, test_size=0.2)

    for itrain, itest in sss.split(df, df[['CLASS']]):
        df_train = df.loc[itrain]
        df_test = df.loc[itest]

    return [df_train, df_test]
```

```
[ ] strat_arr = stratified_sampling(df.drop('Name of Student',axis = 1))

    strat_df_train = strat_arr[0]
    strat_df_test = strat_arr[1]
```



```
[ ] plt.figure(figsize=(10,5))

    plt.subplot(1,2,1)

    strat_df_train['CLASS'].hist()

    plt.subplot(1,2,1)

    strat_df_test['CLASS'].hist()

    plt.show()
```

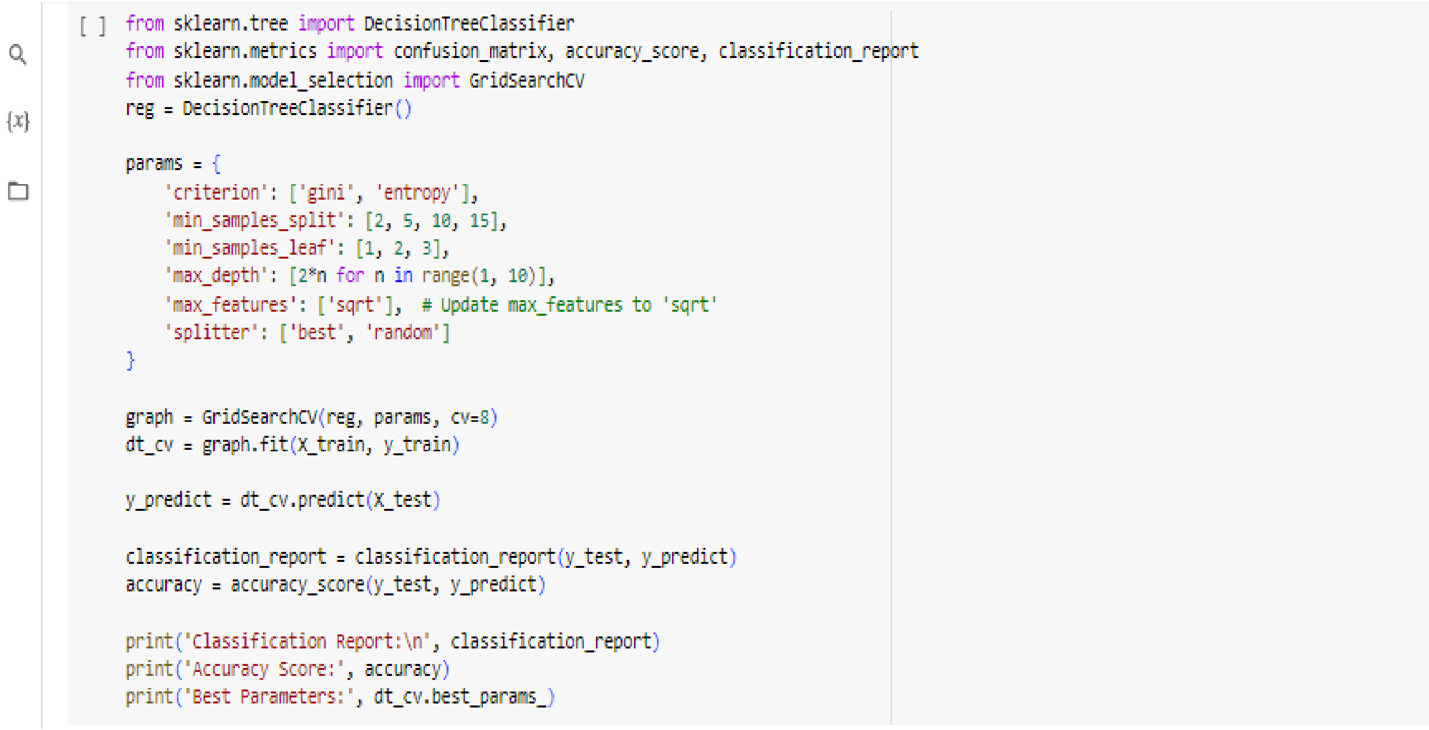


```
[ ] X_train = strat_df_train.drop('CLASS', axis = 1)
    y_train = strat_df_train['CLASS']
```

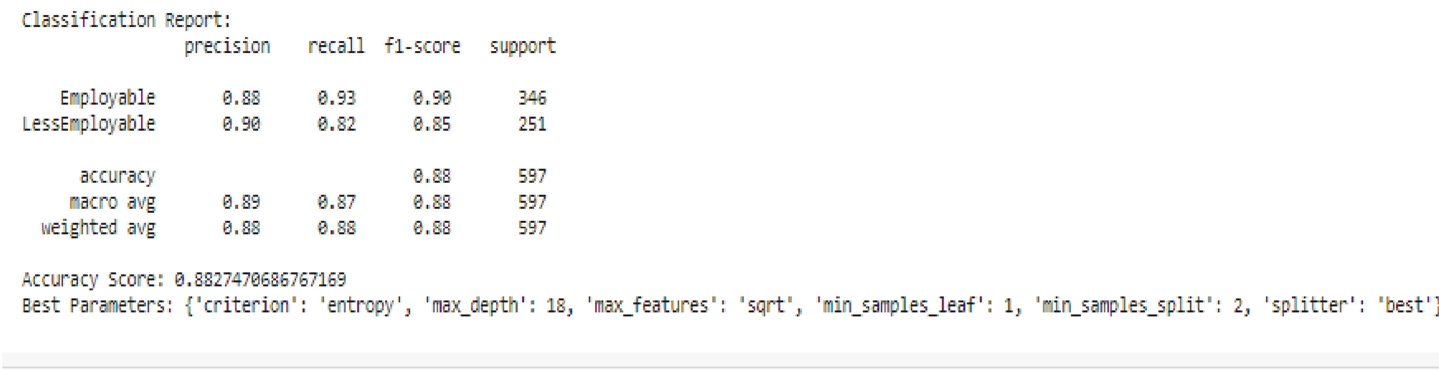
```
[ ] X_test = strat_df_test.drop('CLASS', axis = 1)
    y_test = strat_df_test['CLASS']
```

```
[ ] X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

DECISION TREE CLASSIFIER



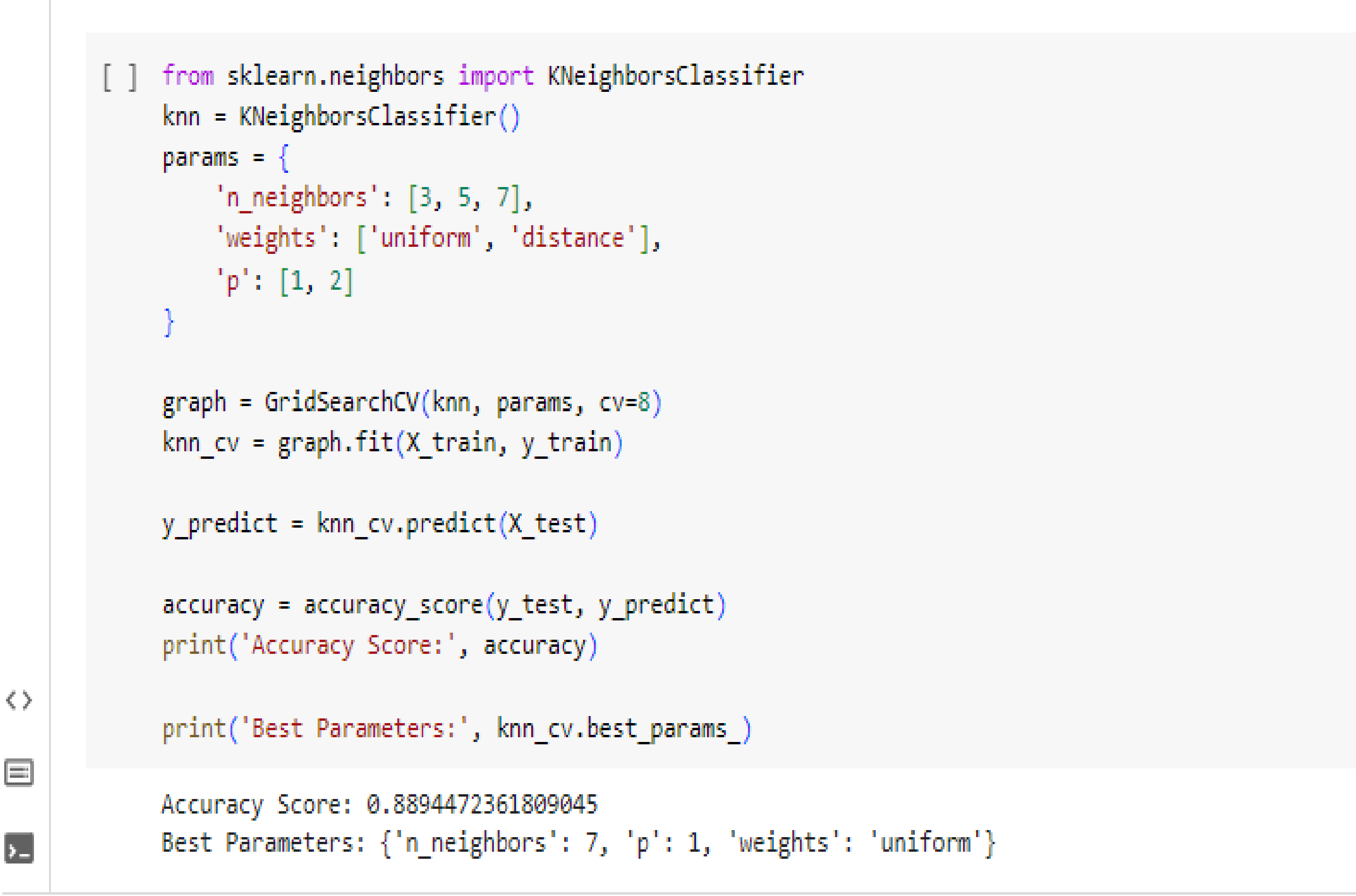
```
[ ] from sklearn.tree import DecisionTreeClassifier
    from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
    from sklearn.model_selection import GridSearchCV
    reg = DecisionTreeClassifier()

    params = {
        'criterion': ['gini', 'entropy'],
        'min_samples_split': [2, 5, 10, 15],
        'min_samples_leaf': [1, 2, 3],
        'max_depth': [2*n for n in range(1, 10)],
        'max_features': ['sqrt'],  # Update max_features to 'sqrt'
        'splitter': ['best', 'random']
    }

    graph = GridSearchCV(reg, params, cv=8)
    dt_cv = graph.fit(X_train, y_train)

    y_predict = dt_cv.predict(X_test)

    classification_report = classification_report(y_test, y_predict)
    accuracy = accuracy_score(y_test, y_predict)

    print('Classification Report:\n', classification_report)
    print('Accuracy Score:', accuracy)
    print('Best Parameters:', dt_cv.best_params_)
```



```
Classification Report:
                precision    recall  f1-score   support

    Employable       0.88      0.93      0.90       346
LessEmployable       0.90      0.82      0.85       251

      accuracy                           0.88       597
     macro avg       0.89      0.87      0.88       597
  weighted avg       0.88      0.88      0.88       597

Accuracy Score: 0.8827470686767169
Best Parameters: {'criterion': 'entropy', 'max_depth': 18, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
```

Accuracy Score of decision tree classifier: 0.8827470686767169

K-NEAREST CLASSIFIER



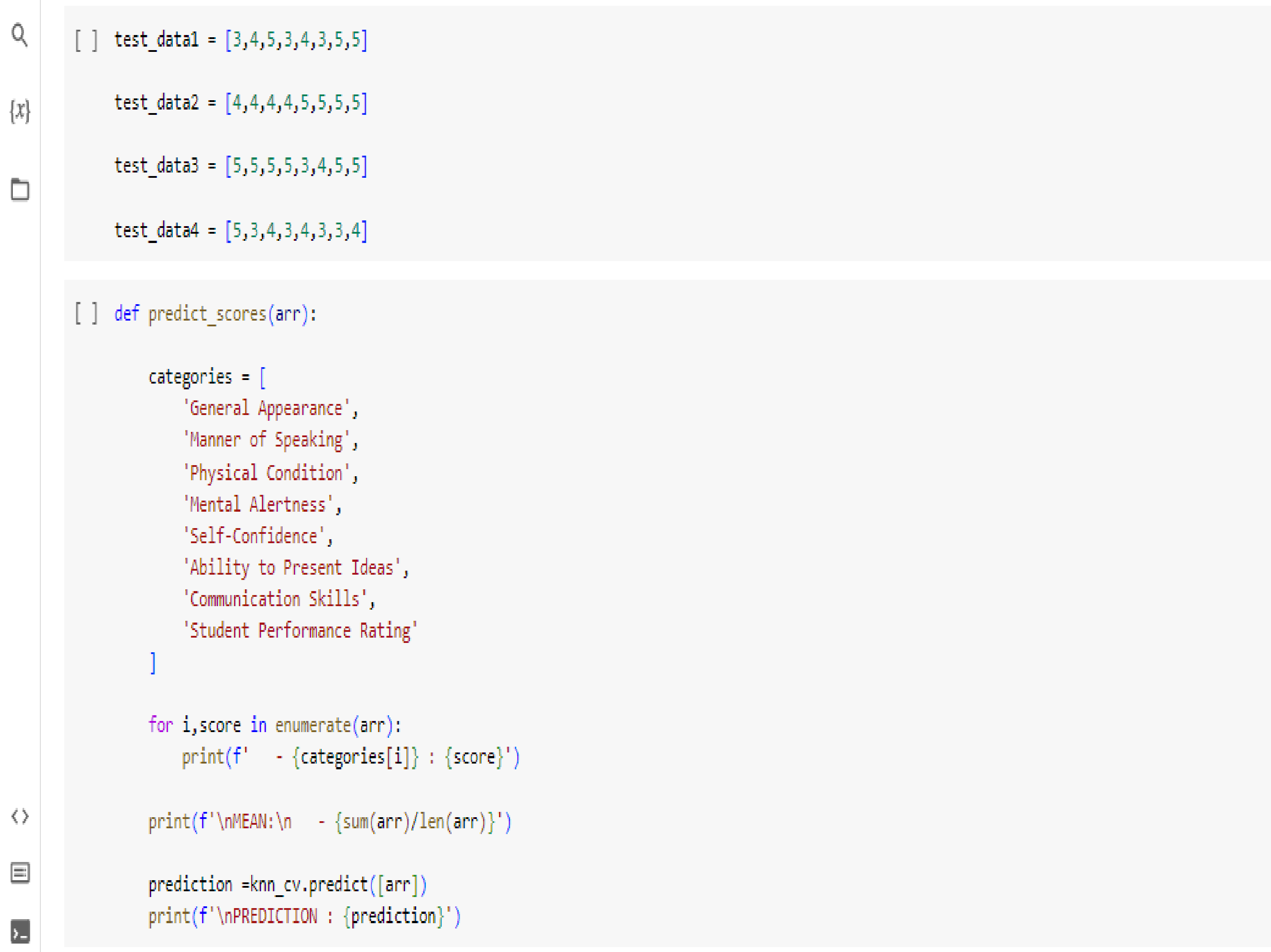
```
[ ] from sklearn.neighbors import KNeighborsClassifier
    knn = KNeighborsClassifier()
    params = {
        'n_neighbors': [3, 5, 7],
        'weights': ['uniform', 'distance'],
        'p': [1, 2]
    }

    graph = GridSearchCV(knn, params, cv=8)
    knn_cv = graph.fit(X_train, y_train)

    y_predict = knn_cv.predict(X_test)

    accuracy = accuracy_score(y_test, y_predict)
    print('Accuracy Score:', accuracy)

    print('Best Parameters:', knn_cv.best_params_)
```

```
Accuracy Score: 0.8894472361809045
Best Parameters: {'n_neighbors': 7, 'p': 1, 'weights': 'uniform'}
```

Accuracy Score: 0.8894472361809045 Which can be rounded off as 0.90

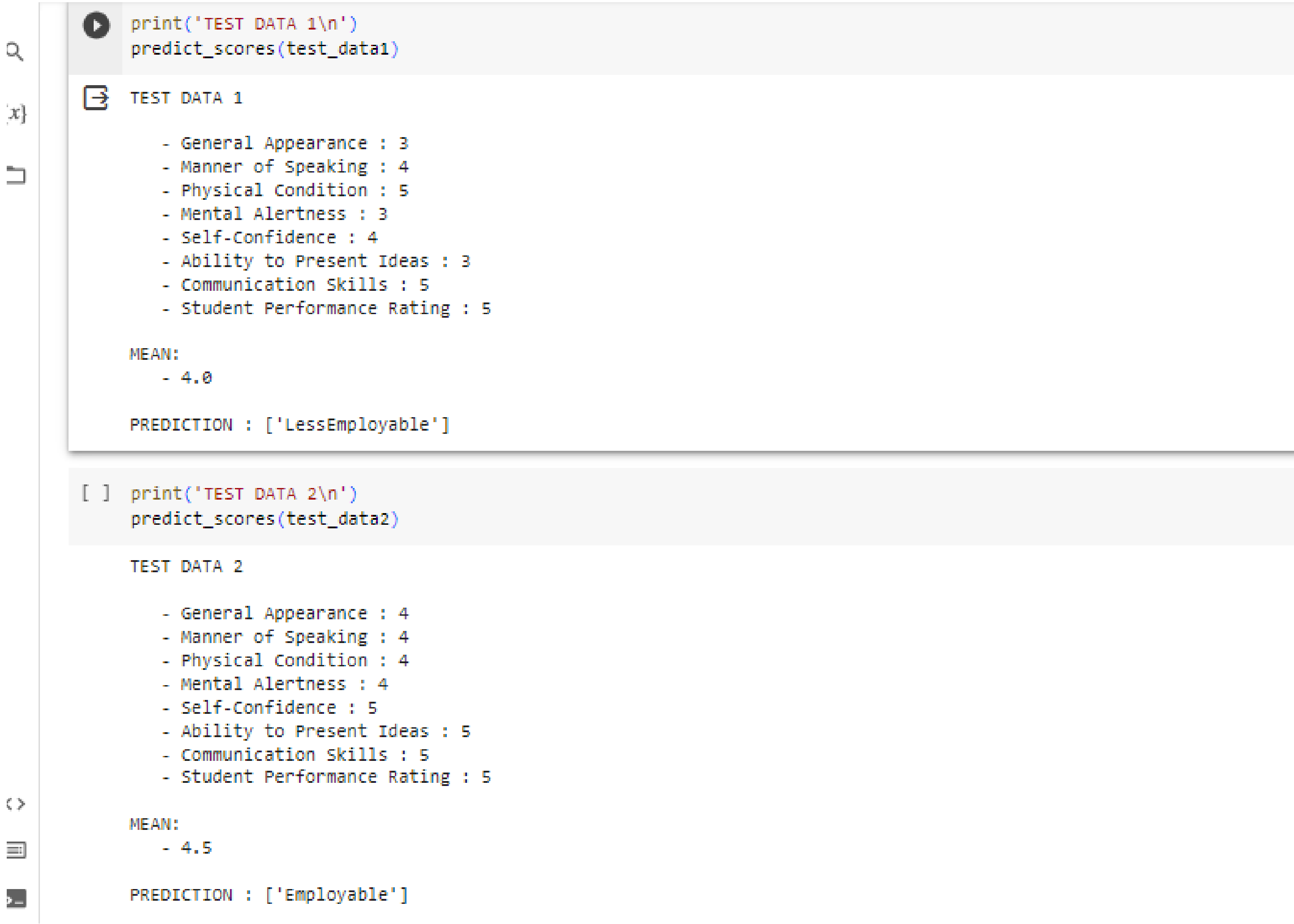
So comparing with decision tree classifier(0.8827470686767169) we can build a model using K-nearest classifier

So We finalize theK-nearest classifier(n_neighbors=7) Model For Our Model Training And Model Deployment



```
test_data1 = [3,4,5,3,4,3,5,5]

test_data2 = [4,4,4,4,5,5,5,5]

test_data3 = [5,5,5,5,3,4,5,5]

test_data4 = [5,3,4,3,4,3,3,4]
```

```
def predict_scores(arr):

    categories = [
        'General Appearance',
        'Manner of Speaking',
        'Physical Condition',
        'Mental Alertness',
        'Self-Confidence',
        'Ability to Present Ideas',
        'Communication Skills',
        'Student Performance Rating'
    ]

    for i,score in enumerate(arr):
        print(f'   - {categories[i]} : {score}')

    print(f'\nMEAN:\n   - {sum(arr)/len(arr)}')

    prediction =knn_cv.predict([arr])
    print(f'\nPREDICTION : {prediction}')
```



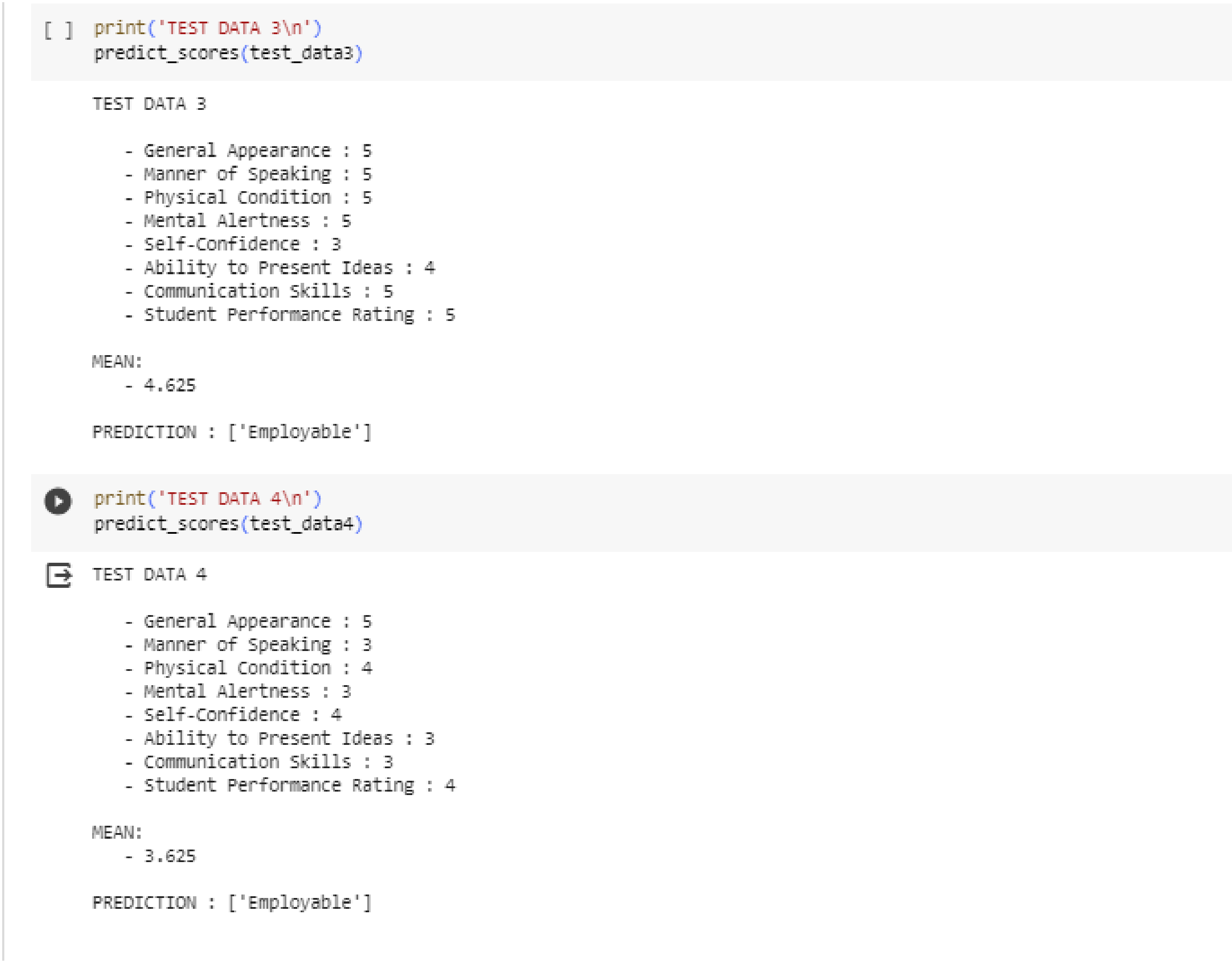
```
print('TEST DATA 1\n')
predict_scores(test_data1)
```

```
TEST DATA 1

   - General Appearance : 3
   - Manner of Speaking : 4
   - Physical Condition : 5
   - Mental Alertness : 3
   - Self-Confidence : 4
   - Ability to Present Ideas : 3
   - Communication Skills : 5
   - Student Performance Rating : 5

MEAN:
   - 4.0

PREDICTION : ['LessEmployable']
```

[ ]

```
print('TEST DATA 2\n')
predict_scores(test_data2)
```

```
TEST DATA 2

   - General Appearance : 4
   - Manner of Speaking : 4
   - Physical Condition : 4
   - Mental Alertness : 4
   - Self-Confidence : 5
   - Ability to Present Ideas : 5
   - Communication Skills : 5
   - Student Performance Rating : 5

MEAN:
   - 4.5

PREDICTION : ['Employable']
```



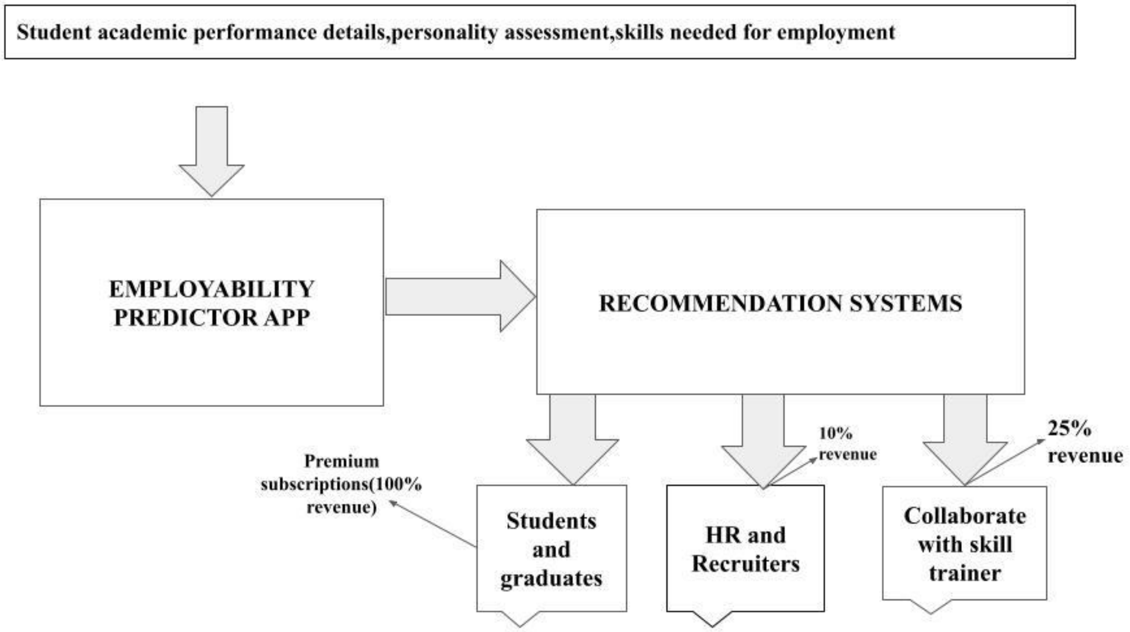
```
[ ] print('TEST DATA 3\n')
    predict_scores(test_data3)

    TEST DATA 3

       - General Appearance : 5
       - Manner of Speaking : 5
       - Physical Condition : 5
       - Mental Alertness : 5
       - Self-Confidence : 3
       - Ability to Present Ideas : 4
       - Communication Skills : 5
       - Student Performance Rating : 5

    MEAN:
       - 4.625

    PREDICTION : ['Employable']
```

```
print('TEST DATA 4\n')
predict_scores(test_data4)

TEST DATA 4

   - General Appearance : 5
   - Manner of Speaking : 3
   - Physical Condition : 4
   - Mental Alertness : 3
   - Self-Confidence : 4
   - Ability to Present Ideas : 3
   - Communication Skills : 3
   - Student Performance Rating : 4

MEAN:
   - 3.625

PREDICTION : ['Employable']
```

11.CONCEPT DEVELOPMENT

The “Employability predictor app” will assess student’s employability using machine learning models that consider academic performance ,skills, and extracurricular activities to provide personalized employability scores and recommendations for skill development ,career pathways, and job market insights.The concept can be developed by using The appropriate API (flask in this case) and using Django as framework for the same and for its deployment , The cloud services has to be chosen accordingly to the need.

12.FINAL PRODUCT PROTOTYPE


Student academic performance details,personality assessment,skills needed for employment
EMPLOYABILITY PREDICTOR APP
RECOMMENDATION SYSTEMS
Premium subscriptions(100% revenue)
Students and graduates
HR and Recruiters
10% revenue
Collaborate with skill trainer
25% revenue

The product takes the following functions to perfect and provide a good result.

Back-end

Model Development: This must be done before releasing the service. A lot of manual supervised machine learning must be performed to optimize the automated tasks.

1. Performing EDA to realize the dependent and independent features.

2. Algorithm training and optimization must be done to minimize overfitting of the model and hyperparameter tuning.

Front End

1. Different user interface: The user must be given many options to choose form in terms of parameters. This can only be optimized after a lot of testing and analysis of all the edge cases.

2. Interactive visualization of the data extracted from the trained models will return raw and inscrutable data. This must be present in an aesthetic and an "easy to read" style.

3. Feedback system: A valuable feedback system must be developed to understand the customer's needs that have not been met. This will help us train the models constantly.

13.PRODUCT DETAILS

How does it work:

- **Users input their academic data,skills, and extracurricular activities,Machine learning models analyze the data to calculate an employability score**
- **Personalized recommendations for skill development and career paths are provided**

Data Sources:

- **Kaggle's "Students Employability Dataset.",**
- **Real-time job market data from online sources**

Algorithms, Frameworks, Software, etc.:

- **Python for machine learning model development**
- **Machine learning libraries like scikit-learn**

- Web development frameworks for building the app's interface

Team Required:

- Data scientists and machine learning engineers, web and app developers, UI/UX designers, database administrators

Cost Estimate:

- Development costs, including salaries for the team, server and hosting expenses, marketing and promotion budget

14. Code Implementation:

https://colab.research.google.com/drive/111M59TUaOI7kvMxIlsT-IXOiBxyP7IqN?usp=sharing

15. Conclusion:

The "Employability Predictor" app aims to address the critical need for personalized career guidance in today's competitive job market. By leveraging machine learning, it provides actionable insights and recommendations to students and job seekers, empowering them to make informed decisions about their education and career paths. The app's potential for monetization and the growing demand for such services make it a promising venture in the education and career counseling sector.