

## **Project on HR Competency Score for Screening using Machine Learning Model**

- **Aim** – Context recruitment and candidate selection can play a critical role in determining the success of an organization. An effective initial screening process can significantly improve the quality of the hiring pool and increase the chances of finding the right candidate for any given role. This project focused on both behavioral and functional competency scores, which are essential aspects of a candidates potential fit and contribution to the organization.
- **Steps to be taken in the Project is sub-divided into the following sections. These are:**
  - Load the necessary libraries such as Numpy , Pandas , sklearn.model etc.
  - Loading the dataset as csv file and showing first five rows.
  - Data preprocessing or (Data cleaning) performed by the one hot encoding in this process we change categorical data into numerical data and the technique is called feature Engineering.
  - Visualization of HR call for Interview by Functional competency and skill score using Tableau.
  - Splitting the cleaned data into dependent and independent variables.
  - Splitting the data into train and test sets with train\_test\_split using sklearn library.
  - Import different kind of Classification Models and Train that model with the help of .fit().
  - Predicting the trained models and then checking their accuracy score and confusion metrics of the model using confusion metrics & accuracy score.
  - Then recall the train\_test\_split and split the data into training and testing set with different models.
  - Then predicting the trained models and checking the accuracy of model and check the accuracy difference.
  - And finally predict whether the HR call for interview classification generated or not.

Shahjan

Khan

## **Step-1** – Loading Necessary Libraries used in machine learning.

```
Loading Necessary Libraries

[ ] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

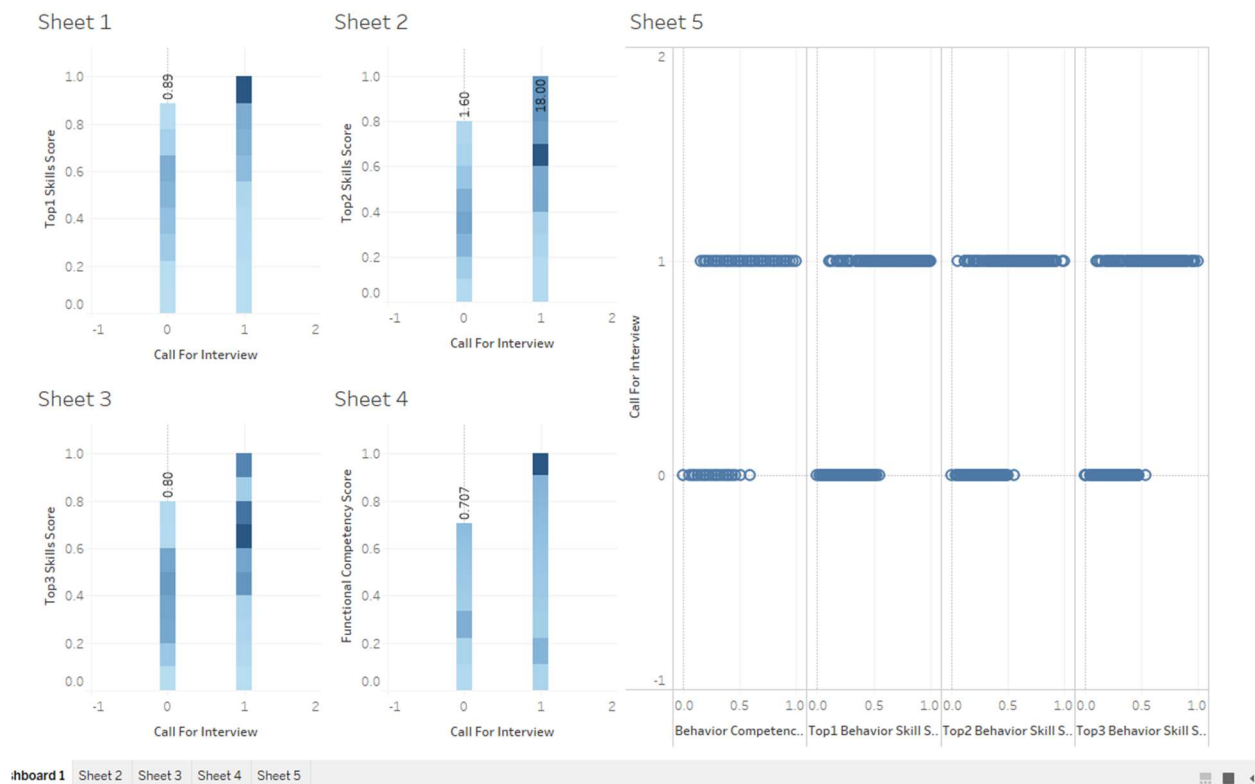
## **Step-2** – Loading and Showing the first five rows of the dataset with .head().

```
Load and show Dataset

[ ] data=pd.read_csv("../content/HR_call.csv")
data.head()
```

years_of_experience	functional_competency_score	top1_skills_score	top2_skills_score	top3_skills_score	behavior_competency_score	top1_behavior_skill_score	top2_behavior_skill_score	top3_behavior_skill_score	call_for_interview
0.333333	0.749498	0.777778	0.7	0.8	0.769231	0.931689	0.662042	0.752463	1
0.133333	0.490638	0.777778	0.3	0.4	0.153846	0.382093	0.132231	0.316905	0
0.000000	0.460256	0.444444	0.3	0.3	0.051282	0.052347	0.089765	0.254859	0
0.000000	0.507347	0.555556	0.4	0.5	0.384615	0.309913	0.289758	0.440784	0
0.400000	0.662020	0.666667	0.7	0.5	0.461538	0.497929	0.397544	0.392760	1

## **Step-3** – Visualization of HR call for Interview by Functional competency and skill score using Tableau.



Shahjan

Khan

**Step-4** – Dividing the data into dependent and independent variables.

deviding the data into dependent(y) and independent(x) variables

```
[ ] x=data.drop(['call_for_interview'],axis=1)
    y=data['call_for_interview']
```

**Step-5** – Splitting the data into train and test sets with train\_test\_split using sklearn library.

Splitting the data into training and testing set

```
[ ] from sklearn.model_selection import train_test_split
    x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.8)
```

**Step-6** – Import first machine learning model K-Nearest neighbor taking n\_neighbor=5.

\_

Creating Machine Learning Model Using K-Nearest Neighbor Model

```
[ ] from sklearn.neighbors import KNeighborsClassifier
    knn=KNeighborsClassifier(n_neighbors=5)
```

**Step-7** – Train the model using .fit() function.

Train the model

```
[ ] knn.fit(x_train,y_train)
```

```
• KNeighborsClassifier
  KNeighborsClassifier()
```

**Step-8** – Predict the trained model using .predict() function.

make predictions on model

```
predictions=knn.predict(x_test)
print(predictions)

[0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 1
 1 0 1 0 1 0 0 1 1 0 1 1 1 0 1 0 1 1 1 1 0 0 0]
```

Shahjan

Khan

**Step-9** – Check the accuracy score and print a confusion metrics with confusion metrics & accuracy score.

```
check the confusion metrics and accuracy score

[ ] from sklearn.metrics import confusion_matrix , accuracy_score
    cm=confusion_matrix(y_test,predictions)
    ac=accuracy_score(y_test,predictions)

[ ] print(ac)

0.9333333333333333

[ ] print(cm)

[[26  0]
 [ 4 30]]
```

**Step-10** - Repeat the process using other machine learning model which is Support Vector Machine (SVM).

```
+ Code + Text Connect ^
create Machine Learning Model Using Support Vector Machine(SVM)

[ ] from sklearn.svm import SVC
    svm_model=SVC()

training the model

[ ] svm_model.fit(x_train,y_train)

+ SVC
SVC()

make predictions on model

[ ] predictions=svm_model.predict(x_test)
    print(predictions)

[0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 1 0 1 1 0 1 1 1 0 1 0 1
 1 0 1 0 1 0 0 1 1 0 1 1 1 0 1 1 1 1 1 1 1 0 0 0]
```

Shahjan

Khan

**Step-11** – Print a confusion metrics and check accuracy score for Support Vector Machine Model.

```
check the confusion metrics and accuracy score

[ ] from sklearn.metrics import confusion_matrix , accuracy_score
    cm=confusion_matrix(y_test,predictions)
    ac=accuracy_score(y_test,predictions)

[ ] print(ac)

0.95

print(cm)

[[26  0]
 [ 3 31]]
```

+ Code + Text

**Step-12** - Repeat the process using other machine learning model which is Logistic Regression and train the model and make predictions.

```
Create Machine Learning Model Using Logistic Regression.

[ ] from sklearn.linear_model import LogisticRegression
    log_model=LogisticRegression()

Training the Model.

[ ] log_model.fit(x_train,y_train)

LogisticRegression
LogisticRegression()

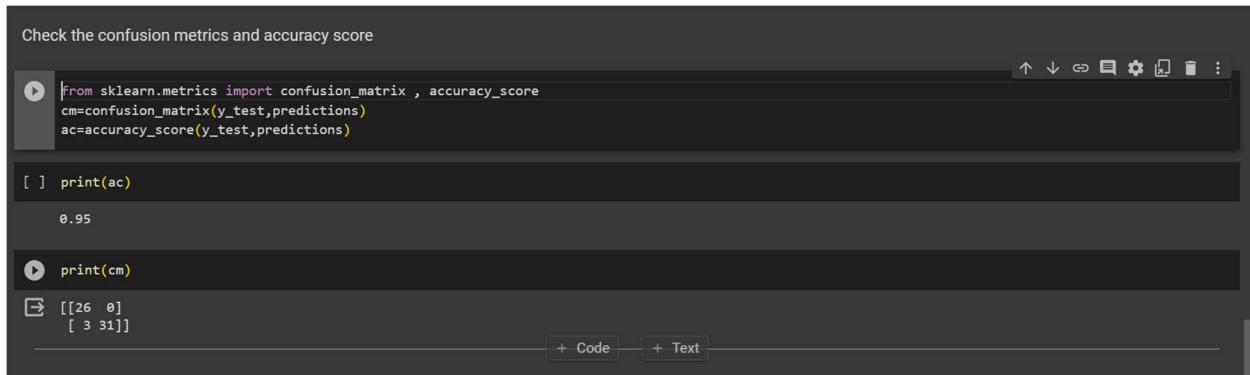
predictions=log_model.predict(x_test)
print(predictions)

[0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 1
 1 0 1 0 1 0 0 1 1 0 1 1 1 0 1 1 1 1 1 1 0 0 0]
```

Shahjan

Khan

**Step-13** – Print a confusion metrics and check accuracy score for Logistic regression Model.



The screenshot shows a Jupyter Notebook interface with a dark theme. The title bar of the cell reads "Check the confusion metrics and accuracy score". The code cell contains the following Python code:

```
from sklearn.metrics import confusion_matrix , accuracy_score
cm=confusion_matrix(y_test,predictions)
ac=accuracy_score(y_test,predictions)

[ ] print(ac)
```

The output of the first code block is the accuracy score: 0.95.

```
print(cm)
```

The output of the second code block is the confusion matrix:

```
[[26  0]
 [ 3 31]]
```

At the bottom of the cell, there are two buttons: "+ Code" and "+ Text".

**Step-14** - Check the accuracies.

1. Using K-Nearest Neighbor – 93% accuracy.
2. Using Support Vector Machine – 95% accuracy.
3. Using Logistic Regression – 95% accuracy.

**Conclusion** – This project is helps to growing need for insights into the hiring process and the importance of selecting candidate who possess a balance of functional and behavioral competencies. This project helps to Data scientist to analyze the initial screening process, build models to optimize candidate selection, explain their decisions, and uncover new insights that can enhance recruitment strategies.

Thank  
you