TASK – 1

TITLE

Identifying potential customers for a fixed deposit subscription using classification algorithms.

1. Introduction

In this part, we will be handling the telemarketing data of Portuguese banking institution which contains the data of the customer's background, their existing loans with the bank, bank balance and if they have any defaults and if they have subscribed for the fixed deposit or not. (Darya,2022) banks use the data collected for various aspects such as Fraud deduction, personalization, cyber security improvements, and automation of banking tasks. The objective of this classification algorithm is to make a model that could predict if a customer would be subscribing to a fixed deposit scheme in the bank, based on the telemarketing data. We will also try to make assumptions about who the target customers are by running a classification algorithm and finding which feature in the dataset has shared a higher relationship with the prediction.

The two major questions to which we try to find a solution are

I. if a customer will be subscribing to a fixed deposit scheme or not.
II. What are the characteristics of the potential group that could subscribe to the fixed deposit scheme?

At the end of the classification algorithm, we will be able to predict if a customer will subscribe to the fixed deposit scheme or not. And we will be able to find a few characteristics of a potential customer. Based on this model the bank can also channel its marketing strategies according to the target audience. And help them to make changes in the scheme to reach a wider set of customers.

2. Dataset

The dataset used in this classification algorithm is downloaded from the prescribed data repository **UCI** under the name **bank marketing**. The dataset contains 45211 instances with 16 variables and an output variable (if the person has subscribed to the fixed deposit or not). The dataset contains the values of the variables as categorical, numeric, and binary data. The data set contains a wide range of data of customers from their age to their banking details which will enable the model to have multiple strong predictor variables and the result of the classification algorithm will not be based on a single predictor variable which can improve the quality of the predictions made. Since we have variables of multiple types, we will be

able to not only predict if the person will subscribe to the plan but also, we will be able to track the characteristic that a potential customer has. We can come to know about these characteristics by investigating the variable and their influence on the prediction made by the classification algorithms These insights will be valuable to fix the target customers and improving the marketing strategies. (Karim., Rahman, 2013) The classification algorithm is being used extensively by companies that need direct marketing to steam line their marketing goals and data mining helps the companies to know their target audience, and this increases the response quality.

3. Explanation and preparation of datasets

The data set has the shape of (45211, 17). The first 16 variables can be used to predict the 17th column which is the output column. And there were no missing values in the dataset.

Data dictionary

Column name	Data type	Description
Age	Numeric	Info of the age of the customer
Job	categorical	info of the job of the customer
Marital status	categorical	Marital status of the customer
Education	categorical	Educational qualification of the customer
default	binary	Info of If the customer has any credit default
housing	binary	Info of if the customer has any hosing loan
loan	binary	Info of if the customer has any personal loan
contact	categorical	Communication mode used
month	categorical	Contact month
Day of week	numeric	Contact day
duration	numeric	Duration of the call
campaign	numeric	Number of contacts made
p-day	numeric	Days elapsed after contact day
previous	numeric	Number of contacts before the campaign
balance	numeric	Bank balance of the customer
poutcome	categorical	Outcome of previous contact
Y	binary	Output variable – if the customer subscribed to the fixed deposit or not

When we are handling public data, we must comply with all licenses and restrictions related to the usage of the specific data. Yet some datasets are made publicly available on a few data repositories for further research and development, and some have restrictions on how the data

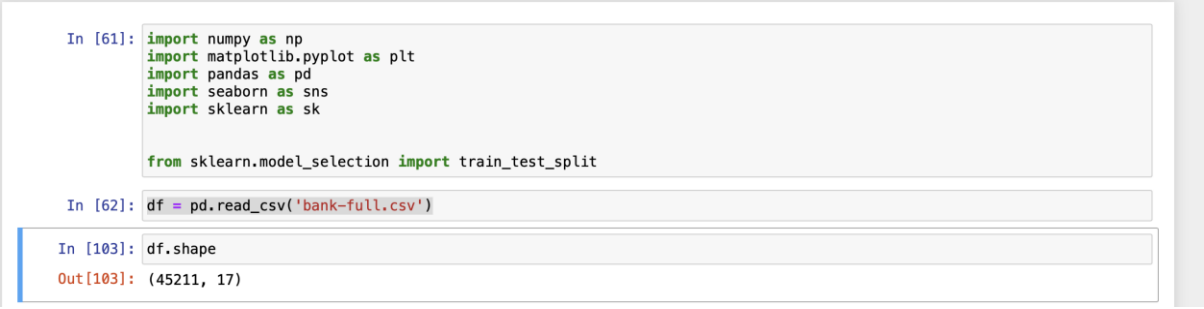
could be used. [Moro et al., 2014] The data used in this research was donated to UCI data in 2012 for public usage and to use for future research and development. Hence the usage of this data is completely legal and ethical. There is no precaution or measures needed to be followed for data privacy since it is public data, which is available to everyone. and it is important to cite the source when we use the dataset.

- Data preparation

(Maneesha,2018) Data pre-processing is an integral part of making an effective classification model. It is a process of transforming raw data into a meaningful full dataset which is understandable in a better and more effective way. The major steps of data pre-processing are importing the libraries which will be needed for conducting the desired experiments with the data, checking for missing values, checking for categorical data, standardizing the data and data splitting.

The libraries which will be used in this task will be as follows:

Figure – 1.1



And the data was imported using the **pd.read_csv** function. And the dimension of the data can be known by using the function **.shape**

Checking for missing values:

Figure – 1.2

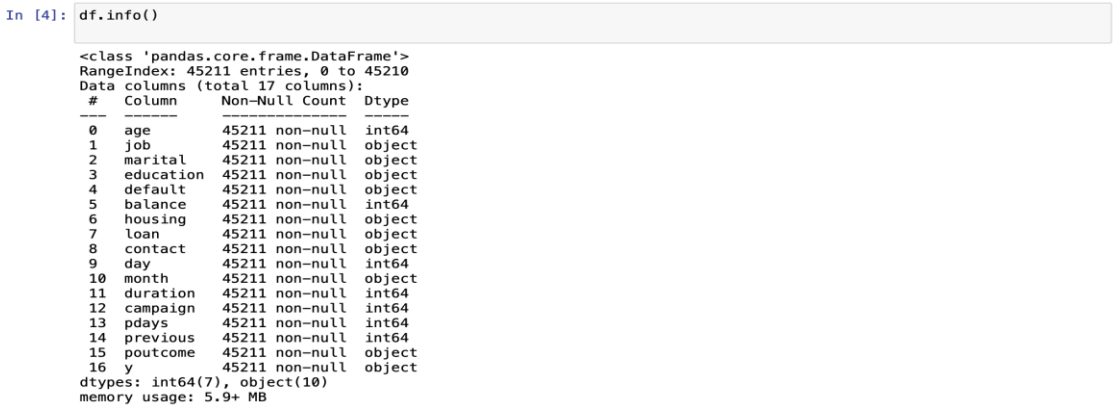


Figure – 1.3

```
In [5]: df.isna().sum()

Out[5]: age          0
        job          0
        marital      0
        education    0
        default      0
        balance      0
        housing      0
        loan         0
        contact      0
        day          0
        month        0
        duration     0
        campaign     0
        pdays        0
        previous     0
        poutcome     0
        y            0
        dtype: int64
```

Using. info() and .isna().sum() we could find if there are any missing values in the dataset. In this case, there are no missing values in the dataset and no missing value techniques need to be used.

Checking for categorical data:

Figure – 1.4

```
In [8]: df.select_dtypes('object')
```

Out[8]:

	job	marital	education	default	housing	loan	contact	month	poutcome
0	management	married	tertiary	no	yes	no	unknown	may	unknown
1	technician	single	secondary	no	yes	no	unknown	may	unknown
2	entrepreneur	married	secondary	no	yes	yes	unknown	may	unknown
3	blue-collar	married	unknown	no	yes	no	unknown	may	unknown
4	unknown	single	unknown	no	no	no	unknown	may	unknown
...	...	...	...	...	...	...	...	...	...
45206	technician	married	tertiary	no	no	no	cellular	nov	unknown
45207	retired	divorced	primary	no	no	no	cellular	nov	unknown
45208	retired	married	secondary	no	no	no	cellular	nov	success
45209	blue-collar	married	secondary	no	no	no	telephone	nov	unknown
45210	entrepreneur	married	secondary	no	no	no	cellular	nov	other

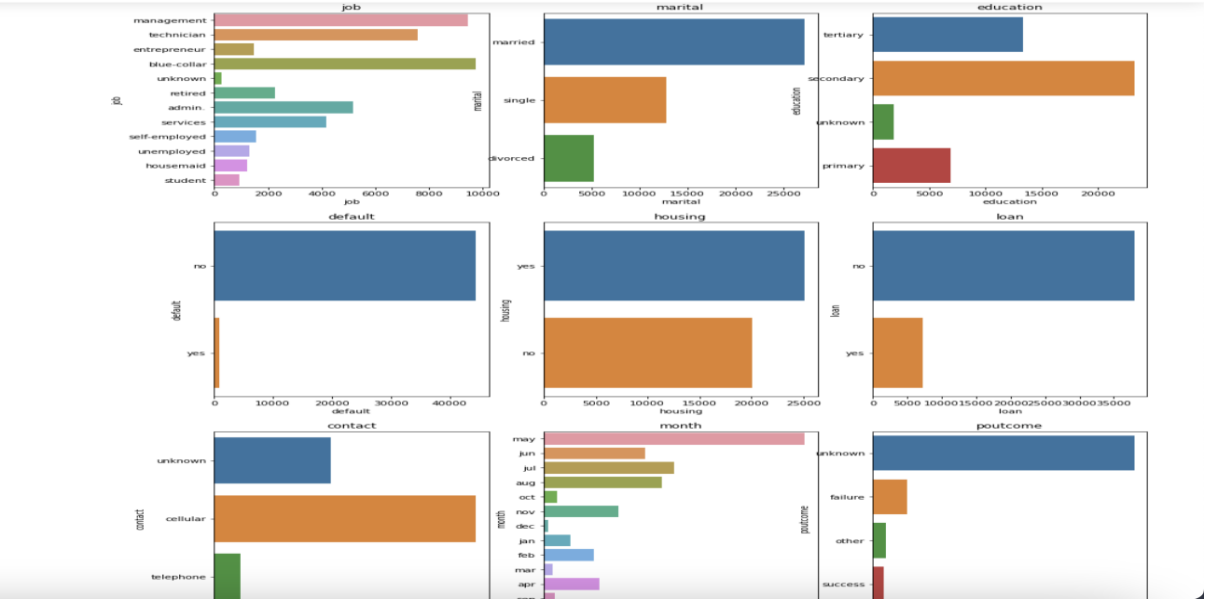
45211 rows × 9 columns

```
In [9]: {column: list(df[column].unique()) for column in df.select_dtypes('object').columns}

Out[9]: {'job': ['management',
           'technician',
           'entrepreneur',
           'blue-collar',
           'unknown',
           'retired',
           'admin.',
           'services',
           'self-employed',
           'unemployed',
           'housemaid',
           'student'],
         'marital': ['married', 'single', 'divorced'],
         'education': ['tertiary', 'secondary', 'unknown', 'primary'],
```

Categorical features:

Figure – 1.5

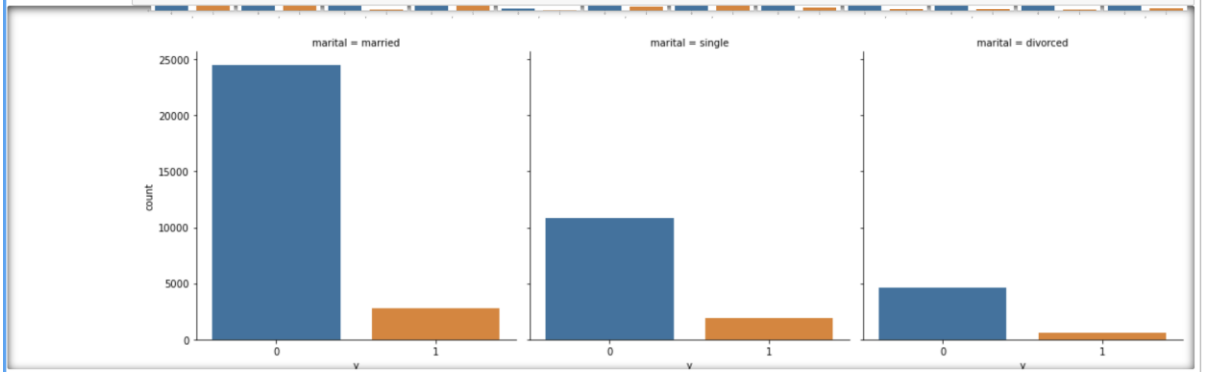
```
In [11]: categorical_features=[feature for feature in df.columns if ((df[feature].dtypes=='O') & (feature not in ['y']))]
         categorical_features

Out[11]: ['job',
          'marital',
          'education',
          'default',
          'housing',
          'loan',
          'contact',
          'month',
          'poutcome']

In [12]: for feature in categorical_features:
             print('The feature is {} and number of categories are {}'.format(feature,len(df[feature].unique())))

         The feature is job and number of categories are 12
         The feature is marital and number of categories are 3
         The feature is education and number of categories are 4
         The feature is default and number of categories are 2
         The feature is housing and number of categories are 2
         The feature is loan and number of categories are 2
         The feature is contact and number of categories are 3
         The feature is month and number of categories are 12
         The feature is poutcome and number of categories are 4

In [13]: plt.figure(figsize=(15,80), facecolor='white')
         plotnumber =1
         for categorical_feature in categorical_features:
             ax = plt.subplot(12,3,plotnumber)
             sns.countplot(y=categorical_feature,data=df)
             plt.xlabel(categorical_feature)
             plt.title(categorical_feature)
             plotnumber+=1
```



We can also find the value of the outcome variable at different selections of categorical values. It can help us have an idea about how strong the variable category is related to the outcome variable. For example, in this dataset, we could see that when the job of a customer is retired the chances of the customer subscribing to the fixed deposit is greater than the customer not subscribing to the fixed deposit.

Figure – 1.6

```
In [14]: for categorical_feature in categorical_features:
             sns.catplot(x='y', col=categorical_feature, kind='count', data= df)
         plt.show()
```



4. Implementation

- **Decision tree**

A decision tree is supervised learning used for classification and regression models. A decision tree is a non-parametric method of learning. Decision tree classification is used to make predictions on the outcome variable using decision rules observed on the features of the data. The decision tree classification method is used widely due to its easy-to-understand and interpretation nature, and it demands a minimum amount of data preparation and preprocessing while other methods use intensive preprocessing methods like using dummy values. A decision tree is capable of handling problems which have multiple output variables.

With all the user-friendly and easy-to-create capabilities, decision tree classifications have their negatives too. (Towards data science, 2018) Smaller changes made in the data can cause a disruption in the model as changes impact the whole structure of the decision tree, decision tree take a longer time to train the model than other classification models.

In this model, we have taken 10 high-performing categorical features to predict the outcome variable. (if the customer subscribed to the fixed deposit or not.). OneHotEncoder was used to transform the high-performing categorical features into a dummy feature (numeric) for effectively training the classification model. And the output variable was converted into a binary variable before splitting the training set and test set. The testing and the training data were split into 20% and 80% respectively.

Figure – 1.6

```
In [33]: df['y'] = df['y'].apply(lambda y: 1 if y == 'yes' else 0)
```

```
In [34]: df
```

Out[34]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	0
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	0
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	0
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	0
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
45206	51	technician	married	tertiary	no	825	no	no	cellular	17	nov	977	3	-1	0	unknown	1
45207	71	retired	divorced	primary	no	1729	no	no	cellular	17	nov	456	2	-1	0	unknown	1
45208	72	retired	married	secondary	no	5715	no	no	cellular	17	nov	1127	5	184	3	success	1
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17	nov	508	4	-1	0	unknown	0
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular	17	nov	361	2	188	11	other	0

45211 rows × 17 columns

Figure 1.7

```
In [45]: x = dataset.iloc[:, 0:10].values
         y = dataset.iloc[:, 11].values
```

```
In [56]: from sklearn.preprocessing import OneHotEncoder
```

```
In [47]: enc_onehot = OneHotEncoder()
         x=enc_onehot.fit_transform(dataset[['job','default','contact','marital','education', 'housing', 'loan','month']]).to
```

```
In [48]: from sklearn.model_selection import train_test_split
         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
```

Every dataset normally has features of multiple dimensions and scales. When the data is of different scales and dimensions it can possibly affect the model to an extent. And also

classification algorithms perform more effectively when the features or scaled to a standard range. The training data is scaled by subtracting the mean and then dividing it by the standard deviation. Fit_transform () and transform () methods were used in this task.

Figure – 1.8

```
In [49]: from sklearn.preprocessing import StandardScaler
         sc=StandardScaler()
         x_train_s=sc.fit_transform(x_train)
         x_test_s=sc.transform(x_test)
```

Now we are trying to import the decision tree classifier from the SKlearn library to make a decision tree model with our dataset and train it. After the training of data is completed, we use the pred function to make predictions with our test data. We can also print the y_test data which is the original output variable in our dataset and compare it with the predicted value.

Figure – 1.9

```
In [50]: from sklearn.tree import DecisionTreeClassifier
         classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
         classifier.fit(x_train_s, y_train)
Out[50]: DecisionTreeClassifier(criterion='entropy', random_state=0)

In [51]: y_pred=classifier.predict(x_test_s)
         print(y_pred)
         [0 0 0 ... 0 0 0]

In [57]: print(y_test)
         [0 1 0 ... 0 1 0]
```

Now we should evaluate the model's performance we will be using the classification report and the confusion matrix. Which tells us about the true positive, false negative, false positive, and true negative values and also the metrics such as precision, f-1 score and recall.

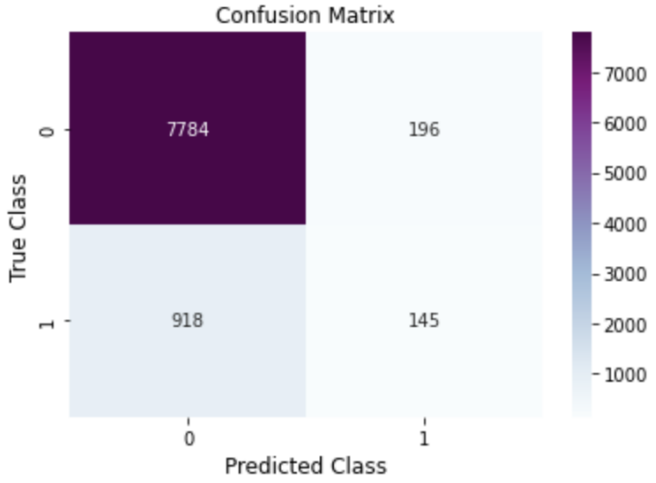
Figure – 1.10

```
In [55]: from sklearn import metrics
         acc=metrics.accuracy_score(y_test,y_pred)
         print('accuracy:%.2f\n\n'%(acc))
         cm=metrics.confusion_matrix(y_test,y_pred)
         print('Confusion Matrix:')
         print(cm,'\n\n')
         print('-------------------------------------------------------')
         result=metrics.classification_report(y_test,y_pred)
         print('Classification Report:\n')
         print(result)
         accuracy:0.88


         Confusion Matrix:
         [[7784  196]
          [ 918  145]]


         -------------------------------------------------------
         Classification Report:

                       precision    recall  f1-score   support

                    0       0.89      0.98      0.93      7980
                    1       0.43      0.14      0.21      1063

             accuracy                           0.88      9043
            macro avg       0.66      0.56      0.57      9043
         weighted avg       0.84      0.88      0.85      9043
```

Figure – 1.11

```
In [53]: from sklearn.metrics import plot_confusion_matrix
```

```
In [54]: ax = sns.heatmap(cm, cmap = 'BuPu',  annot=True, fmt='d')

         plt.xlabel("Predicted Class", fontsize=12)
         plt.ylabel("True Class", fontsize=12)
         plt.title("Confusion Matrix", fontsize=12)

         plt.show()
```



- **KNN**

Knn is also non-parametric and supervised learning similar to a decision tree classifier. Knn makes use of the closeness between the data points to make a prediction. (Harrison, 2018) It makes predictions to which group a particular data belongs to by investigating the data point and checking which group is the nearest from the data point. The uniqueness of the classifier is it does not make use of the training data and stores it. It doesn't perform any calculations until there is a query.

Knn is a very versatile and easy-to-use classifier. It can be used for a wide range of tasks. But Knn's speed is directly proportional to the size of the independent variables. It is faster when there are fewer independent variables and gets slower when there are more independent variables. data preparation, preprocessing and all other steps are common between knn and decision tree. hence we will start by importing the KNN classifier from the sklearn library.

Figure – 1.12

```
In [23]: from sklearn.neighbors import KNeighborsClassifier
         classifier=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)
         classifier.fit(x_train_s,y_train)

Out[23]: KNeighborsClassifier()
```

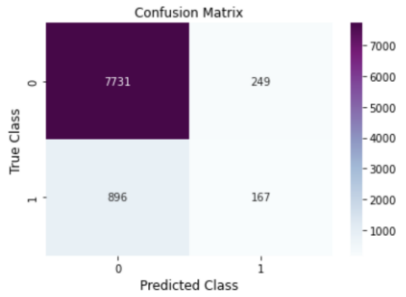
The results of the KNN classifier models are as stated below:

Figure – 1.13

```
In [25]: from sklearn import metrics
         acc=metrics.accuracy_score(y_test,y_pred)
         print('accuracy:%.2f\n\n'%(acc))
         cm=metrics.confusion_matrix(y_test,y_pred)
         print('confusion Matrix:')
         print(cm,'\n\n')
         print('--------------------')
         result=metrics.classification_report(y_test,y_pred)
         print('classification')
         print(result)

accuracy:0.87


confusion Matrix:
[[7731  249]
 [ 896  167]]


--------------------
classification
              precision    recall  f1-score   support

           0       0.90      0.97      0.93      7980
           1       0.40      0.16      0.23      1063

    accuracy                           0.87      9043
   macro avg       0.65      0.56      0.58      9043
weighted avg       0.84      0.87      0.85      9043
```

Figure – 1.14

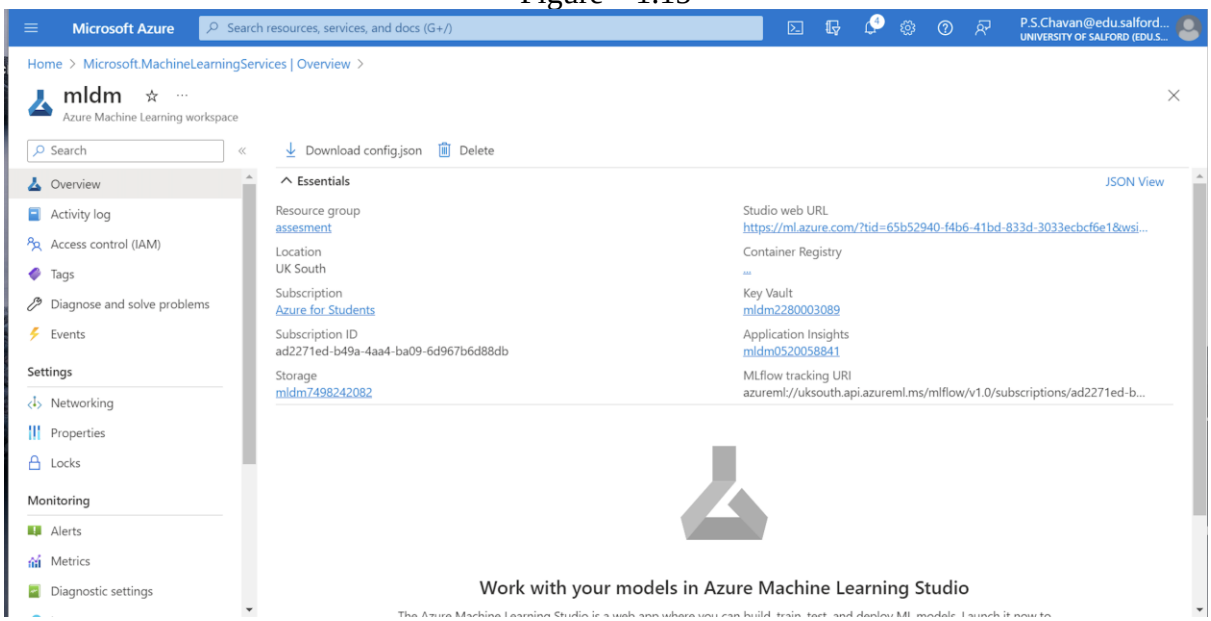
```
In [26]: from sklearn.metrics import plot_confusion_matrix

In [27]: ax = sns.heatmap(cm, cmap = 'BuPu',  annot=True, fmt='d')

         plt.xlabel("Predicted Class", fontsize=12)
         plt.ylabel("True Class", fontsize=12)
         plt.title("Confusion Matrix", fontsize=12)

         plt.show()
```



- **Microsoft AZURE**

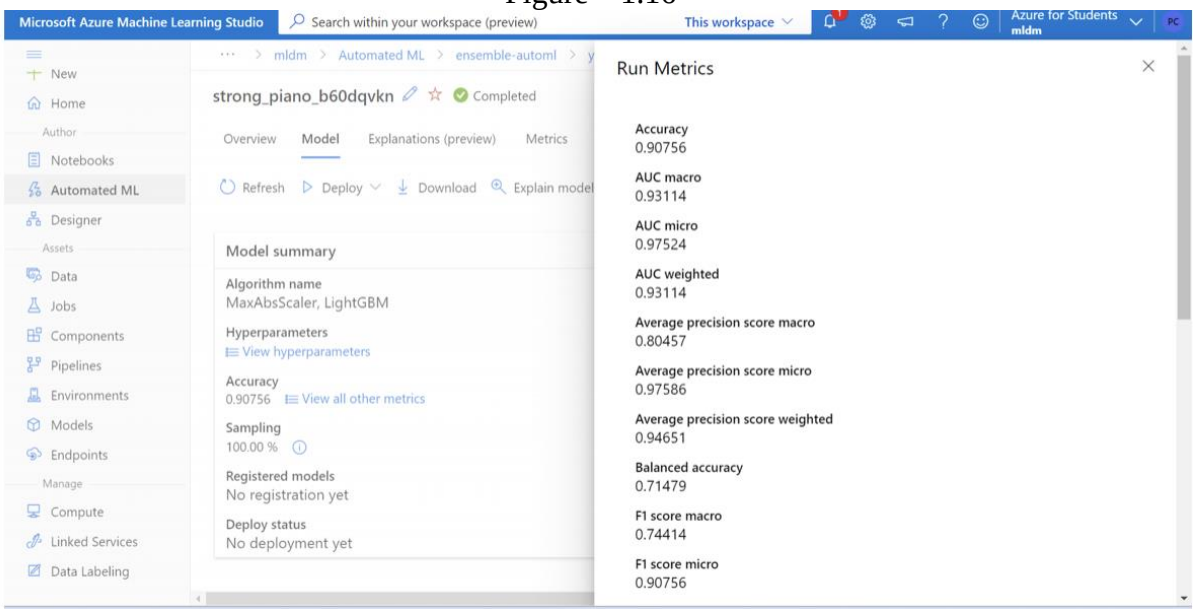
Using of Microsoft azure helps us to perform multiple tasks using different algorithms at the same time it gives us an idea about how each algorithm treats a dataset and what is the accuracy of each algorithm without the use of coding. It enables us to compare the algorithms and select an algorithm which fits the best for our objective. We have run our data set through various classification algorithms in this part. We were able to do this by creating an azure machine-learning workspace.

Figure – 1.15



When we ran our data set through various algorithms were able to find that maxabsSclaer, Light GBM was the best fit for the dataset with an accuracy of 0.9075630.

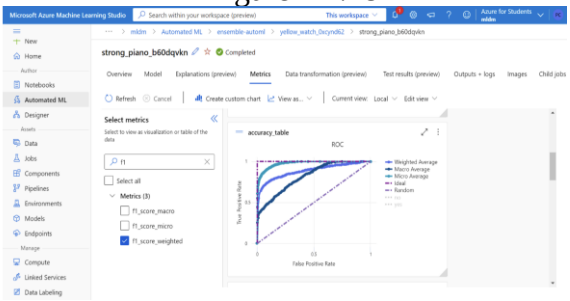
Figure – 1.16



it also provides us with different types of metrics and visualization to have a better understanding.

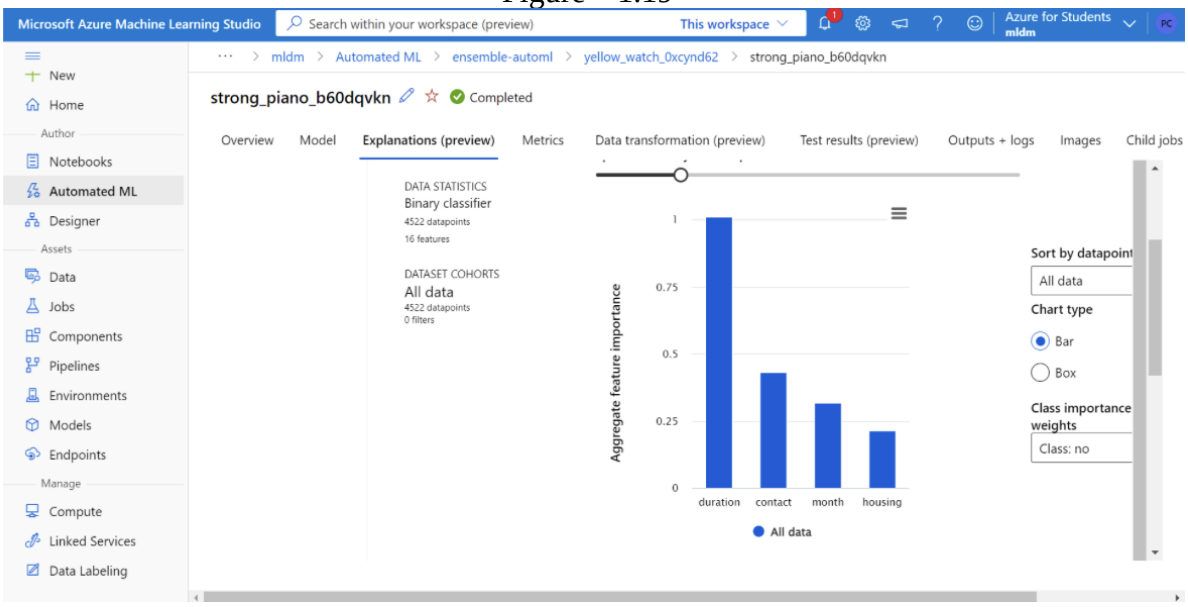
Figure – 1.17.

Figure – 1.18

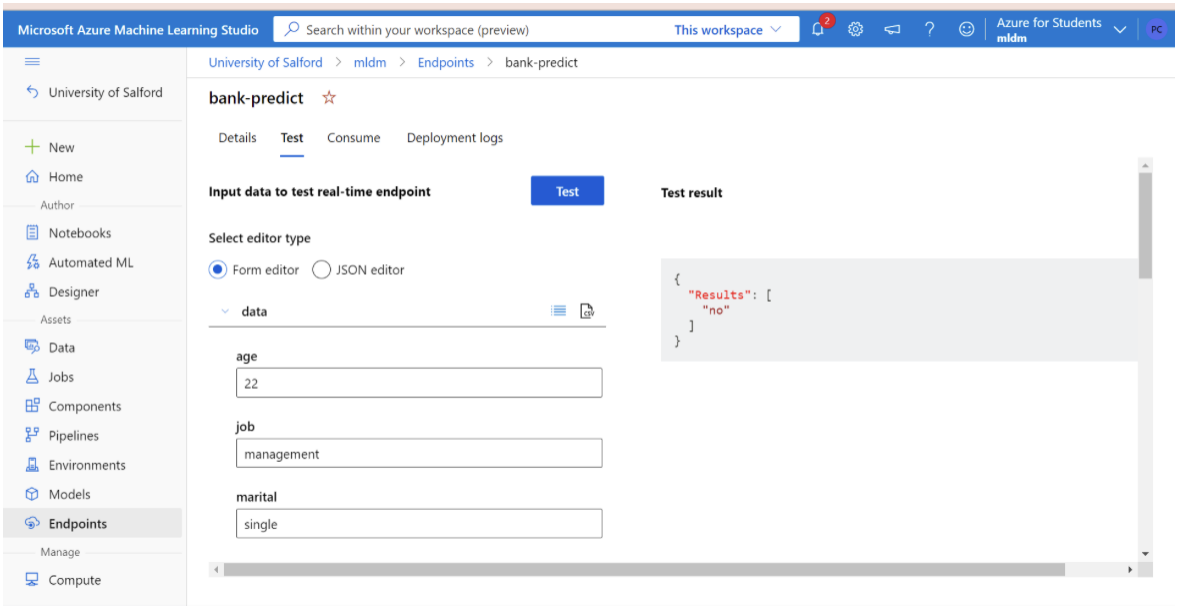


We can also get the feature importance in an order

Figure – 1.19



Using Microsoft Azure, we can deploy a predictive model and test it with different values and check its prediction in real-time.



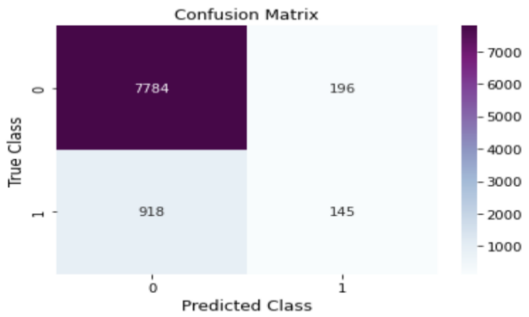
- Result analysis

We will be using different metrics to compare the performance of the models such as F1 score, confusion matrix and accuracy.

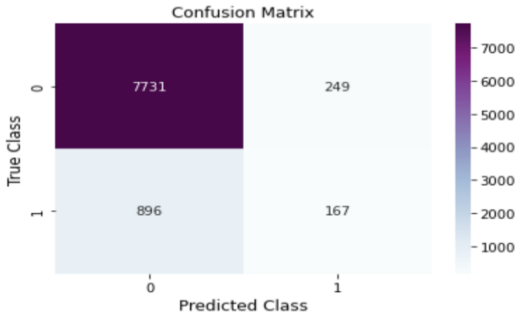
Classifier	F1 score	Accuracy
Decision tree	0.93	0.88
KNN	0.93	0.87

en we compare the F1 and accuracy of both the models we can see that there is no chance in the F1 score but there is a slight improvement in the accuracy in the decision tree than KNN.

Decision tree



KNN



Classifier	True positive	True negative	False positive	False negative
Decision tree	7784	145	196	918
KNN	7731	167	249	896

(Agarwal,2022) When we compare all three metrics, we are able to find that there isn't any significant between the classifier, but we could conclude that the decision tree has performed slightly better than KNN overall and decision tree was the best fit for the dataset.

During the data mining process, we were able to find some valuable insights that could solve research questions. We found that retired people are inclined more to get a fixed deposit than people with other jobs. It could help us to know a potential customer group. Marketing can be more focused on retired people to get better results. And the strategies followed, and the elements of the fixed deposit can also be fine tunned while knowing our target audience better. We also found that the longer the call duration the higher the chance of the customer to subscribe for the fixed deposit.

Overall, we were able to make a classification model with an accuracy of 0.88 and we were also able to find some key characteristics of a potential customer group which was also one of our research questions.

- **Conclusion**

The main objectives of this assignment were:

I. if a customer will be subscribing to a fixed deposit scheme or not.
II. What are the characteristics of the potential group that could subscribe to the fixed deposit scheme?

We were able to make predictive models using various classification algorithms and we were successfully able to make a model with 0.88 accuracies. This model will help us to know (predict) if a customer will be subscribing to the fixed deposit scheme or not.

And during the various stages of the data mining process, we were able to find key insights into the characteristics of the potential customer group that could subscribe to the fixed deposit scheme. We were able to find that retired people are more inclined into getting a fixed deposit scheme. We were able to solve all the research questions set at the beginning of