Customer_Age – This column holds customer age in years.

Gender – This column holds gender of the customer which is either M = Male or F = Female

Dependent_count – This column holds number of dependents

Education_Level – This column holds highest education level of the customer which is Uneducated,High School,Graduate,Post-Graduate,Doctorate.

Marital_Status – This column holds marital which can be Married or Single or Divorced

Income_Category – This column holds annual income category of the customer which is Less than $40K, $80K - $120K, $60K - $80K, $60K - $80K, $120K +.

Card_Category – This column holds type of card. Platinum cards offer more reward pointsthan gold cards, which offers more reward points than silver cards. Blue cards offer the least number of reward points on purchases.

Months_on_book – This column holds period of relationship (in months) with bank.

Total_Relationship_Count – This column holds total number of products held by the customer (e.g., Saving account, Car loan, Credit Card, etc.)

Months_Inactive - This column holds number of months inactive in the last 12 months

Contacts_Count – This column holds number of customer service contacts in the last 12 months

Credit_Limit – This column holds credit limit on the credit card

Total_Revolving_Bal – This column holds total revolving balance on the credit card (the portion of credit card spending that goes unpaid at the end of a billing cycle)

Total_Trans_Amt – This column holds total transaction amount in the last 12 months

Total_Trans_Ct – This column holds total transaction count in the last 12 months

Attrition_Flag – This column holds two labels - ‘Existing Customer’, ‘Attrited Customer’ (Customer who has churned)

1.Data Preparation (What steps would you take to prepare your data? Discuss your approach

i. Importing Libraries,Functions: For data preparation, I have imported the necessary libraries and functions. These include pandas, sklearn, imblearn, read_csv, get_dummies, DataFrame, StandardScaler, Fit_transform, train_test_split, and SMOTE.
ii. Importing Dataset: The dataset is named customerchurn.csv, and I have saved this file on my Google Drive. By connecting Google Colab to my drive, I have mounted it and gained access to the file. I have obtained the file path, used the read_csv

function in dataset variable to read the file, and then have printed the dataset to examine its contents.

iii. Data Quality Check:In the dataset we have checked that is there any outlier which means is there any null value,missing value in the dataset.

iv. Data Encoding: I have used Two types of Encoding
    - Label Encoding- In this encoding either there are two unique values or values which we can order or rank.The values which can be ranked for that features I have used label encoding.Attrition_Flag,Gender,Education_Level,Income_Category,Card_Category
    - One Hot Encoding-This encoding is used for the features which cannot be ranked .We have used one hot encoding for 'Marital_Status'.It contains 3 unique values which cannot be ranked or ordered.

v. Scaling: The given dataset is scaled to bring all the discrete values in a particular range.We have scaled the dataset using Standard_scaler function.

vi. Splitting: In this step we are dividing the dataset into training set and testing set.We are using train_test_split function to divide the dataset.We are using 70% of dataset for training and the remaining 30% for testing.

2. Model Hyperparameter Tuning (Which hyperparameters would you tune and why? How would you tune them?)

Random Forest Classifier- The Random Forest Classifier is a classification algorithm that utilizes multiple decision trees trained on different subsets of a given dataset. It improves the accuracy of predictions by taking the average of the results obtained from each individual tree. Rather than relying on a single decision tree, the random forest combines the predictions of all the trees and determines the final output based on the majority vote among the predictions.

A Random Forest Classifier has several Hyperparameters that can be changed :

i. N_estimators :This is the number of trees you want to build,before taking the average of predictions.

ii. Criterion: The method by which a decision tree's split quality is evaluated. Following are supported criteria 'gini' for the gini impurity and 'entropy' for information gain.

iii. Max_features: This parameter determines the maximum number of features that can be considered when splitting a node in a random forest. It helps in finding the optimal split. There are four possible values for this parameter: 'auto', 'sqrt', 'log2', and 'none'.

iv. Random State: It is random selection of index from the dataset while building a tree.

v. Max_depth: The number of columns that are shown to each decision tree.

Which hyperparameters would you tune and why? How would you tune them?

i. Criterion : Value used entropy, Entropy is a measure of disorder or impurity in a node. A node with a more diverse composition, such as having a mix of two "Pass" and two "Fail" instances, would be considered to have higher entropy compared to a node that consists of only "Pass" or only "Fail" instances. Higher entropy indicates a greater level of uncertainty or variability within the node.
ii. N_estimators: The value of n_estimator is auto picked by using GridsearchCV,we just have to provide the list of n_estimators. Higher the number of trees better the performance ,and as a result the predictions are accurate as well.
iii. Max_features: Value used sqrt, which will simply take the square root of all features . The reason to use this hyperparameter is, if you allow all the features for each split you are going to end up exactly the same trees in the entire random forest which might not be useful.

Support Vector Machine : Support Vector Machines (SVM) is a supervised machine learning algorithm used for both classification and regression tasks.The main objective of SVM is to classify datasets by identifying the maximum marginal hyperplane. This hyperplane represents the best line or decision boundary that effectively separates classes in an n-dimensional space, allowing future data points to be accurately categorized.

i. kernels: Kernels encompass a set of pattern analysis algorithms that enable the solution of non-linear problems using a linear classifier. By employing kernel functions, which transform the input data into a higher-dimensional space, these algorithms can effectively address non-linear patterns in the data.
ii. C: It is a regularizing parameter.

Which hyperparameters would you tune and why? How would you tune them?

i. Kerenels: The GridSearchCV function is utilized to select the best kernel for an SVM model. RBF is the kerenel chosed by GridSearchCV.
ii. C:The GridSearchCV function is employed to determine the best value for a specific parameter to enhance the predictions of a model. It takes in a range or set of different values that can be tested to find the optimal value that improves the performance of the model. By systematically evaluating the model's performance with different parameter values, GridSearchCV assists in identifying the most suitable value that leads to improved predictions.

3.Choice of Evaluation Metric (Which metric would be suitable for model evaluation and why?)

i. Accuracy:The accuracy is the number of correctly predicted decisions out of the total number of records/index.The accuracy is calculated as sum of True Positive and True Negative divided by the sum of True Positive True Negative False Positive False Negative.
ii. Precision: It measures the proportion of correctly predicted positive instances out of the total instances predicted as positive.Precision is calculated as the number of True Positive divided by the sum of True Positive and False Positive.
iii. Recall: It measures the proportion of correctly predicted positive instances out of all the actual positive instances in the dataset.Recall is calculated as the number of True Positive divided by the sum of True Positive and False Negative.
iv. Confusion Matrix:It is a NxN matrix,where N is the total number of classes. The Confusion Matrix comprises True Positives, True Negatives, False Positives, and False Negatives, providing insights into the accurate anticipation of actual true positives and negatives.

Which metric would be suitable for model evaluation and why?

I have tested all the metrics,It boiled down to two choices precision and recall. Recall is the number of True Positive divided by the sum of True Positive and False Negative. Precision is the number of True Positive divided by the sum of True Positive and False Positive.For both precision and recall it was showing the almost identical values.The output feature of our dataset is Attrition_Flag which means was the customer an existing customer or attrited customer.Since we have to predict weather he was a existing customer or attrited customer and not the accuracy of the correctness which means model predicting the customer is existing or attrited is more important than percentage of correct result.

4. Overfitting avoidance mechanism (Which mechanism (feature Selection/ regularization) would you use and why?)

Introduction: Overfitting is the phenomenon in which the model correctly predicts predicts the results of training set because it has memorized the training examples,but when it is given a new data set its accuracy decreases,it is not able to predict the correct result.The model is not able to predict correct results of unseen data.Overfitting can happen when a model is to complex to the available data and when there is lot of noise or irrelevant information in the training dataset.

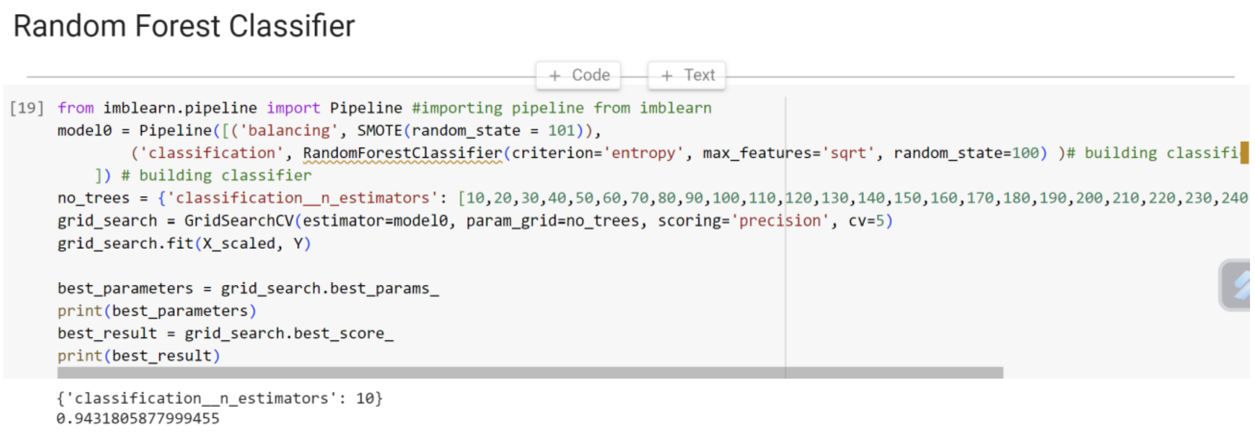
Which mechanism (feature Selection/ regularization) would you use and why?

I have used regularization in my model because it is mainly applied on linear models.It helps to control the complexity of the model. It involves adding a penalties which encourages the model to find a balance between fitting the training data and keeping the model's parameters small.In my model I have used regularization parameter in SVM over a range of 0.001 to 100.In this way we can avoid overfitting.Feature selection is avoided because the dataset is small and does not have multiple features.

5. Results analysis a). Which of the two models (random forest or support vector classifier) would you recommend for deployment in the real-world? b). Is any model underfitting? If yes, what could be the possible reasons?

My approach consists of six steps which are as follow**:**

i. Data Preperation:In this step we have imported all the necessary libraries functions. Converted object data type to either int or float data type by using label encoding or one hot encoding.
ii. Scaling and splitting:When the data is prepared with all the int or float data type we have scaled the dataset,scaling means making all the discrete values in a particular range.After scaling we have splitted the dataset into training and testing dataset we have given 70% dataset to training and 30% of dataset to testing.
iii. Training:For training I have used two algorithms Random Forest Classifier and Support Vector Machine.I have used Precision evaluation for both the algorithms.The result of Random Forest Classifier using Precision is .The result of Support Vector Machine using Precision is .
iv. Testing: I have used Precision evaluation for both the algorithms.The result of Random Forest Classifier using Precision is 94.31% .The result of Support Vector Machine using Precision is 96.56% .

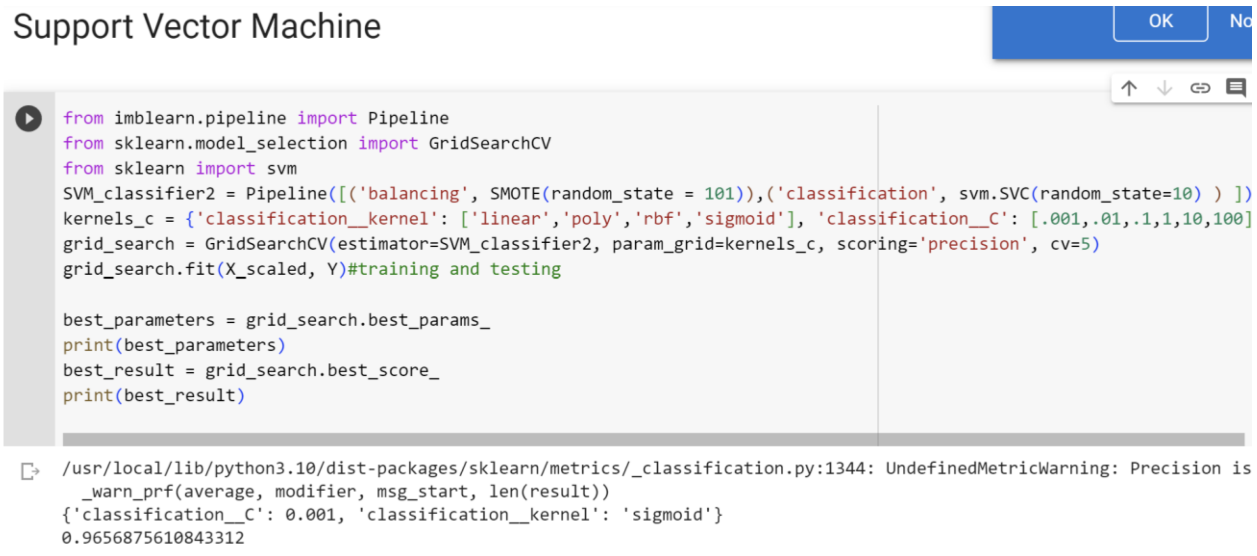


Random Forest Classifier

```
[19] from imblearn.pipeline import Pipeline #importing pipeline from imblearn
     model0 = Pipeline([('balancing', SMOTE(random_state = 101)),
             ('classification', RandomForestClassifier(criterion='entropy', max_features='sqrt', random_state=100) )# building classifi
         ]) # building classifier
     no_trees = {'classification__n_estimators': [10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,170,180,190,200,210,220,230,240
     grid_search = GridSearchCV(estimator=model0, param_grid=no_trees, scoring='precision', cv=5)
     grid_search.fit(X_scaled, Y)

     best_parameters = grid_search.best_params_
     print(best_parameters)
     best_result = grid_search.best_score_
     print(best_result)
```

```
{'classification__n_estimators': 10}
0.9431805877999455
```



```
from imblearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn import svm
SVM_classifier2 = Pipeline([('balancing', SMOTE(random_state = 101)),('classification', svm.SVC(random_state=10) ) ])
kernels_c = {'classification__kernel': ['linear','poly','rbf','sigmoid'], 'classification__C': [.001,.01,.1,1,10,100]
grid_search = GridSearchCV(estimator=SVM_classifier2, param_grid=kernels_c, scoring='precision', cv=5)
grid_search.fit(X_scaled, Y)#training and testing

best_parameters = grid_search.best_params_
print(best_parameters)
best_result = grid_search.best_score_
print(best_result)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision is
  _warn_prf(average, modifier, msg_start, len(result))
{'classification__C': 0.001, 'classification__kernel': 'sigmoid'}
0.9656875610843312
```

Conclusion: After seeing the results of both the algorithm comparing both of them Support Vector Machine results are greater than the Random Forest Classifier. Hence I would recommend Support Vector Machine for deployment in real world.

b). Is any model underfitting? If yes, what could be the possible reasons?

Underfitting means the training dataset is not able to correctly and accurately predict the results .Underfitting means High Bias and low variance.By using both the algorithm none of the model is underfitting.Both Random Forest Classifier and Support Vector Machine are not Underfitting.