PROJECT - 2

PYTHON: NGO-FUND RAISING ATTRITION- CLASSIFICATION

ABSTRACT:

The data at hand is a part of the database of a fundraising organization.

The fundraising organization gave us a data dump as of 02/02/2007. They will use the model to score their customer base in terms of their churn probability.

SUMMARIES OF PROBLEM, DATA, METHODS, AND TECHNOLOGIES:

❖ PROBLEM SUMMARY

The problem in hand can be divided into four pieces namely –

A> Perform required data manipulation and cleaning
B> Dividing data into training/testing time windows.
C> Create models using base table (training data)
D> Asses the performance of your model (validation) by test data

❖ DATA SUMMARY

The input data provided is in SAS data format (sas7bdat). The data need to be imported using 'read_sas' function of 'pandas' library.

Input sas data: \input\

There are total **8** data frames that need to be imported for the solution –

Detailed Data Dictionaries are given below -

1. **Extrel**: All the donors of the organization

Variable	Description
Extrelno	Unique identifier of each donor
Exrelactcd	Activity code of the donor
Extrelstdt	Start date of the relationship
Exreldaten	End date of the relationship (Missing: not ended)

2. **Extrelty**: Description of the activity

Variable	Description
Exrelactcd	Activity code of the donor
Exrelactde	Description of the activity

Email: banerjee.kaustav@outlook.com
Github: https://github.com/KBanerjee90

3. **Nameaddr**: Sociodemographical information

Variable	Description
Extrelno	Unique identifier of each donor
Name1title	Title to address someone
Postcode	Postcode
Languagecode	Preferred mailing language

4. **Payhistory**: Paymenthistory of each donor

Variable	Description
Pid	Unique identifier for each payment
Pdate	Date of payment
Pamt	Amount of payment
Extrelno	Unique identifier of each donor
Paytypecd	Paytype *O Bank transfer* *D Permanent order* *E Own initiative* *X Unknown*
Status	Status of payment *OK Normal/Real payment* *CO Correction (internal)* *RF RF (Refund)* *RC Recall*

5. **Communication**: All possible communication between the donor and the organization

Variable	Description
Contid	Unique identifier for each contact
Mediumcode	Medium of the contact (CI is unknown)
Mntopcode	Main topic code of the contact
Classcode	Class of the contact
Extrelno	Unique identifier for each donor
Contdirec	Direction of the communication *I Incoming* *P Outgoing*
Contdate	Date of the contact

6. **Commediu**: Description of medium type

Variable	Description
Mediumcode	Code of the mediumtype
Mediumdesc	Description

7. Commaint: Description of the main topic code

Variable	Description
Mntopcode	Main topic code
Mntopdesc	Description

8. **Comclas**: Description of the contact class

Variable	Description
Clascode	Code of contact class
Clasdesc	Description

❖ METHODS SUMMARY

Table shows the wide variety of data pre-processing, analysis, and visualization techniques that I applied to complete the tasks as part of the project –

Task ID	Task Details	Analytical Techniques	Visualization Techniques
Data Manipulation and preparation	1> Perform required data manipulation and cleaning. 2> Dividing data into different time windows. 3> Create the Base table (for training models)	Descriptive statistics Straightforward data manipulation	pandas_profiling.Profile Report matplotlib.pyplot seaborn (Heatmap)
Modelling and performance	Create models and Asses the performance of your model	Logistic Regression, Decision Tree and Random Forest	

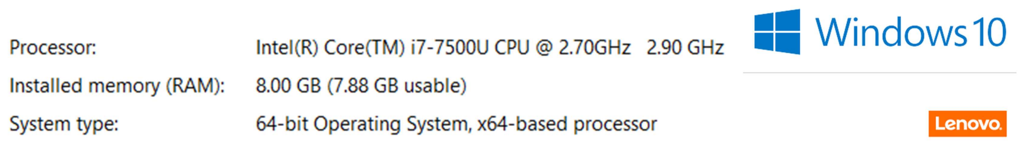
❖ TECHNOLOGIES SUMMARY

Although size was not the most significant difficulty presented by the fundraising organization; but the variation of data was worth special consideration.

The data has lots of cardinality and variation that needs to be handled by data manipulation.

The following list summarizes the technology that I used:

Computing platforms:

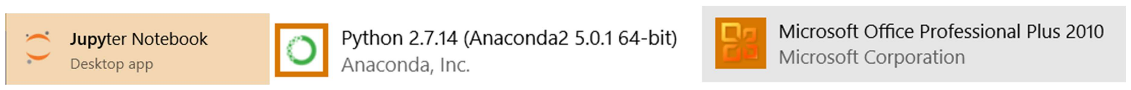


Processor:	Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz
Installed memory (RAM):	8.00 GB (7.88 GB usable)
System type:	64-bit Operating System, x64-based processor

Microsoft Excel 2010 on the single-machine platform

Jupyter Notebook on the single-machine platform

Python 2.7.14 (Anaconda2 5.0.1 64bit on the single-machine platform



Jupyter Notebook
Desktop app

Python 2.7.14 (Anaconda2 5.0.1 64-bit)
Anaconda, Inc.

Microsoft Office Professional Plus 2010
Microsoft Corporation

Email: banerjee.kaustav@outlook.com
Github: https://github.com/KBanerjee90

MODEL TO SCORE CUSTOMER BASE IN TERMS OF THEIR CHURN PROBABILITY

❖ PART A: PERFORM REQUIRED DATA MANIPULATION AND CLEANING

I downloaded the input SAS data in into python data frame.

```
comclas_df = pd.read_sas('comclas.sas7bdat')
commaint_df = pd.read_sas('commaint.sas7bdat')
commediu_df = pd.read_sas('commediu.sas7bdat')
communication_df = pd.read_sas('communication.sas7bdat')
extrel_df = pd.read_sas('extrel.sas7bdat')
extrelty_df = pd.read_sas('extrelty.sas7bdat')
nameaddr_df = pd.read_sas('nameaddr.sas7bdat')
payhistory_df = pd.read_sas('payhistory.sas7bdat')
```

Used straightforward data manipulation and descriptive statistics to perform data manipulation –

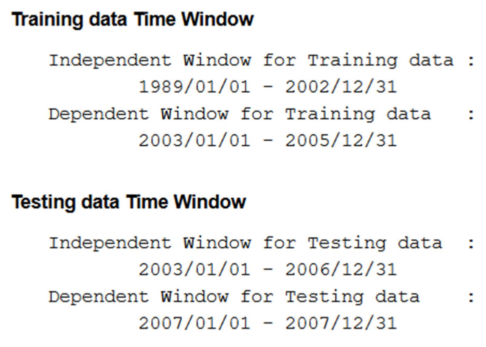
A. Working with Payment History Data
    a. Creating New KPIs Frequency, Recency, Total and average donation per donor and Pay type per customer.
    b. Create new variables that signify whether a donor ever used sendout, order, own initiative and unknown. Creating Dummies for PAYTYPECD of payment.
    c. Roll up data at EXTRELNO Level; so that there is only one observation per customer.
    d. Filtering out the Normal Payment (OK) Only to be used for further processing.
B. Working with Communication History Data
    a. Dummy whether the donor made a complaint.
    b. Dummy whether communication direction was ever incoming.
    c. Roll up data at EXTRELNO Level; so that there is one observation per customer.
C. Working with Nameaddr: Sociodemographical information
    a. Getting Preferred mailing language
    b. Getting Dummies for LANGUAGE CODE - Categorical variable
D. Creating the final data (extrel_df_final) merging the manipulated datasets with extrel_df on key 'EXTRELNO'
E. I did not include all categorical variables because of high cardinality; the categorical variables converted into dummies are as –
   [PAYTYPECD, COMMUNICATION_TYPE, LANGUACODE, CONTDIREC and CLASCODE]

```
nameaddr_by_exrel          --> prefered mailing language details
communication_by_exrel  --> communication details
payhistory_by_extrel       --> payhistory details
```

Email: banerjee.kaustav@outlook.com
Github: https://github.com/KBanerjee90

- **PART B: DEFINING THE TIME WINDOW FOR TRAIN AND TEST DATSETS**

The Data is available from 1989 – 2007.



Training data Time Window

```
Independent Window for Training data :
        1989/01/01 - 2002/12/31
Dependent Window for Training data   :
        2003/01/01 - 2005/12/31
```

Testing data Time Window

```
Independent Window for Testing data  :
        2003/01/01 - 2006/12/31
Dependent Window for Testing data    :
        2007/01/01 - 2007/12/31
```

So, we define an '**active customer**' as one with - activity code as '**FP**' start date before end of Independent Window to end date after start of Dependent window [means he was active during Independent window] or missing [means still a donor].

In this way we have **6313** count for train dataset for model building and **1569** count of test datasets for validation of model.

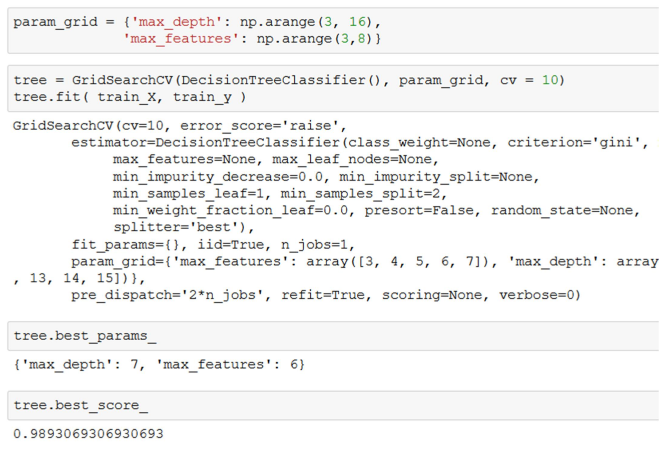
- **PARTC & D: CREATE MODELS USING BASE TABLE AND ASSES THE PERFORMANCE OF YOUR MODEL (VALIDATION) BY TEST DATA**

I used Decision Tree and Random forest to create the model and validate them.

Using Grid search I got the best parameter incurring 99% accuracy as **max_depth = 7, max_features=6.**

Code Snippet –

Grid Search to find the best parameters (code snippet) -



```
param_grid = {'max_depth': np.arange(3, 16),
              'max_features': np.arange(3,8)}
```

```
tree = GridSearchCV(DecisionTreeClassifier(), param_grid, cv = 10)
tree.fit( train_X, train_y )
```

```
GridSearchCV(cv=10, error_score='raise',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini',
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best'),
       fit_params={}, iid=True, n_jobs=1,
       param_grid={'max_features': array([3, 4, 5, 6, 7]), 'max_depth': array
, 13, 14, 15])},
       pre_dispatch='2*n_jobs', refit=True, scoring=None, verbose=0)
```

```
tree.best_params_
```

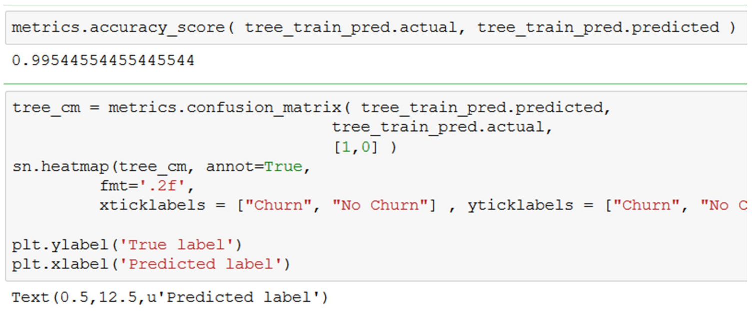
```
{'max_depth': 7, 'max_features': 6}
```

```
tree.best_score_
```

```
0.9893069306930693
```

*Metrics Accuracy – **99%***

Code Snippet and heatmap:-

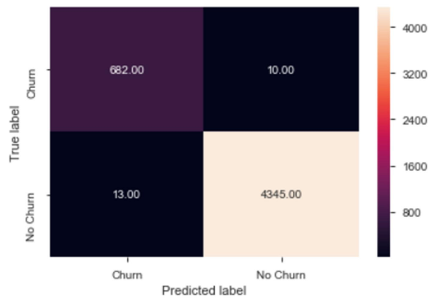


```
metrics.accuracy_score( tree_train_pred.actual, tree_train_pred.predicted )
0.995445544554455445544
```

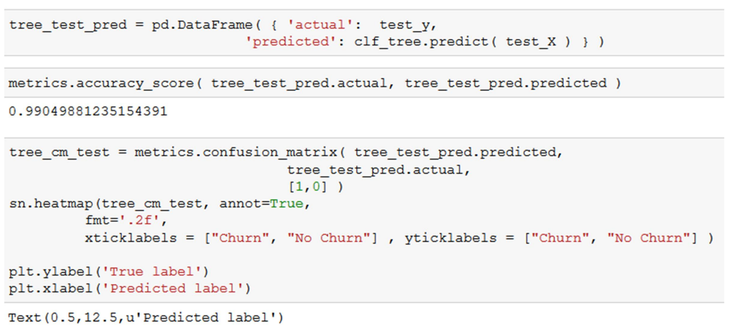
```
tree_cm = metrics.confusion_matrix( tree_train_pred.predicted,
                                    tree_train_pred.actual,
                                    [1,0] )
sn.heatmap(tree_cm, annot=True,
         fmt='.2f',
         xticklabels = ["Churn", "No Churn"] , yticklabels = ["Churn", "No C

plt.ylabel('True label')
plt.xlabel('Predicted label')
```

```
Text(0.5,12.5,u'Predicted label')
```



MODEL VALIDATION



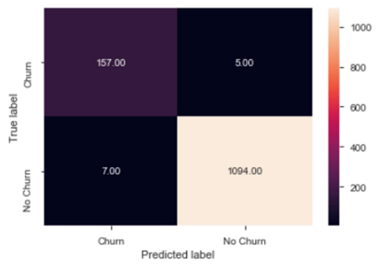
```
tree_test_pred = pd.DataFrame( { 'actual':  test_y,
                                'predicted': clf_tree.predict( test_X ) } )
```

```
metrics.accuracy_score( tree_test_pred.actual, tree_test_pred.predicted )
0.99049881235154391
```

```
tree_cm_test = metrics.confusion_matrix( tree_test_pred.predicted,
                                         tree_test_pred.actual,
                                         [1,0] )
sn.heatmap(tree_cm_test, annot=True,
         fmt='.2f',
         xticklabels = ["Churn", "No Churn"] , yticklabels = ["Churn", "No Churn"] )

plt.ylabel('True label')
plt.xlabel('Predicted label')
```

```
Text(0.5,12.5,u'Predicted label')
```

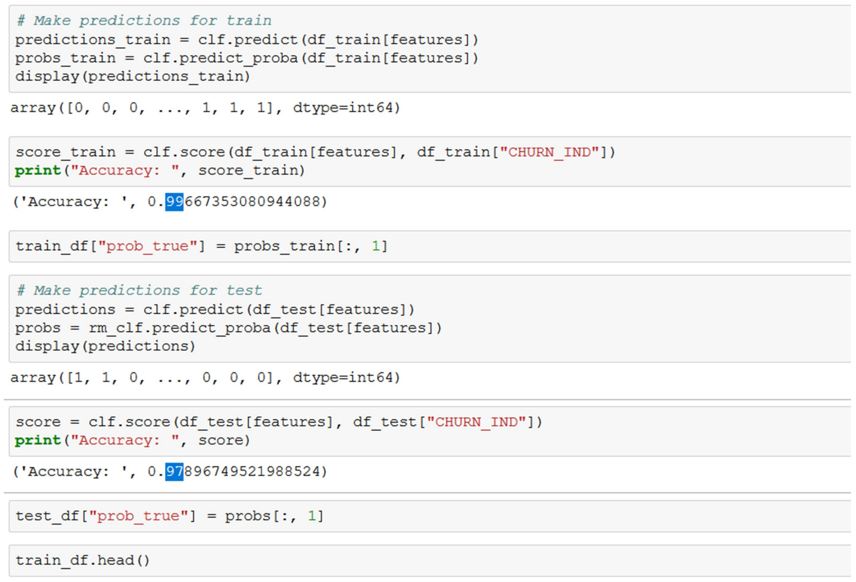


❖ TESTING AND VALIDATION

BUILDING RANDOM FOREST MODEL TO GET THE PROBABILITY OF CHURN FOR CUSTOMERS –

I used Random Forest Classifier ‘predict’ and ‘predict_proba’ to get the prediction probability; we will use that to estimate the probability of churn -

Code Snippet:-



```
# Make predictions for train
predictions_train = clf.predict(df_train[features])
probs_train = clf.predict_proba(df_train[features])
display(predictions_train)
```

```
array([0, 0, 0, ..., 1, 1, 1], dtype=int64)
```

```
score_train = clf.score(df_train[features], df_train["CHURN_IND"])
print("Accuracy: ", score_train)
```

```
('Accuracy: ', 0.99667353080944088)
```

```
train_df["prob_true"] = probs_train[:, 1]
```

```
# Make predictions for test
predictions = clf.predict(df_test[features])
probs = rm_clf.predict_proba(df_test[features])
display(predictions)
```

```
array([1, 1, 0, ..., 0, 0, 0], dtype=int64)
```

```
score = clf.score(df_test[features], df_test["CHURN_IND"])
print("Accuracy: ", score)
```

```
('Accuracy: ', 0.97896749521988524)
```

```
test_df["prob_true"] = probs[:, 1]
```

```
train_df.head()
```

We are able to achive more that 99% accuracy for TRAIN and ~98% accuracy for TEST VALIDATION

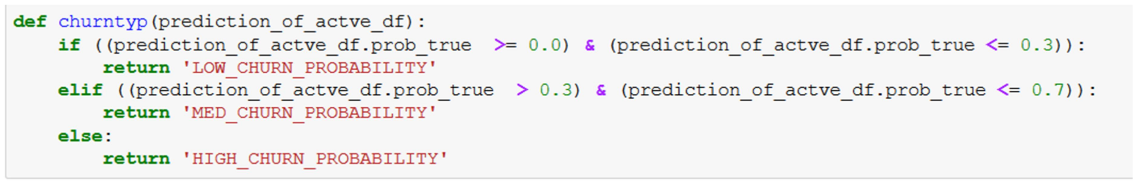
❖ RESULTS

Concatenation of test and train dataset and filtering out the active customers we can able to get the complete result

I divided the churn probabilities into 3 categories namely -

0.0 - 0.3	- LOW_CHURN_PROBABILITY
> 0.3 – 0.7	- MED_CHURN_PROBABILITY
> 0.7	- HIGH_CHURN_PROBABILITY

Code Snippet for BANDING:-



```
def churntyp(prediction_of_actve_df):
    if ((prediction_of_actve_df.prob_true  >= 0.0) & (prediction_of_actve_df.prob_true <= 0.3)):
        return 'LOW_CHURN_PROBABILITY'
    elif ((prediction_of_actve_df.prob_true  > 0.3) & (prediction_of_actve_df.prob_true <= 0.7)):
        return 'MED_CHURN_PROBABILITY'
    else:
        return 'HIGH_CHURN_PROBABILITY'
```

Email: banerjee.kaustav@outlook.com
Github: https://github.com/KBanerjee90

Based on the band grouping we get the count per BAND as -

CHURN BAND	Count of EXTRELNO
HIGH_CHURN_PROBABILITY	16
LOW_CHURN_PROBABILITY	5541
MED_CHURN_PROBABILITY	1453
Grand Total	**7010**

Result output - output/ Fundraising-CHURN-Prediction-Active-With-BAND.xlsx

Solution code - Solution/ NGO-FUND_RAISING_ATTRITION-CLASSIFICATION_Solution.ipynb

RECOMMENDATIONS

For customers with **High Churn probabilities** will require an immediate call to understand their grievances/complains and reason for non-payment/inactivity.

May be a new relationship manager will be helpful to understand their choices in terms of payment, language and complaints.

For customers with **Medium Churn probabilities** –may require **recognition or souvenirs** for their contribution to NGO.

Keep in touch with call and do monitor their grievances and complains for better contribution.

REFERENCES

https://github.com/

https://www.kaggle.com/

Analytixlabs class 14, example Case Study - HR Analytics

Email: banerjee.kaustav@outlook.com
Github: https://github.com/KBanerjee90
Linkedin: https://www.linkedin.com/in/kaustav-banerjee-584525149