'Smart Adjustments': A Machine Learning Approach to Automate Adjustment procedures in General Ledgers.

A

Project Report

submitted in partial fulfillment of the

requirements for the award of the degree of

MASTER OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND ARTIFICIAL NEURAL NETWROK

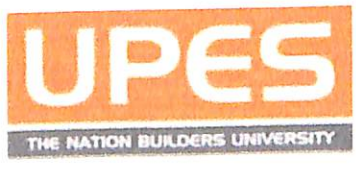
by

Name	Roll No.
Glen Bennet Hermon	**R102214003 (SAP:500036672)**

Under the guidance of

Mr. Gunjan Chhabra

Centre of Information Technology, UPES, Dehradun



Department of Computer Science & Engineering

Centre for Information Technology

University of Petroleum & Energy Studies

Bidholi, Via Prem Nagar, Dehradun, UK

April – 2016

The innovation driven
E-School

CANDIDATE'S DECLARATION

I hereby certify that the project work entitled "**'Smart Adjustments': A Machine Learning Approach to Automate Adjustment procedures in General Ledgers.**" in partial fulfilment of the requirements for the award of the Degree of MASTER OF TECHNOLOGY in COMPUTER SCIENCE ENGINEERING with specialization in ARTIFICIAL INTELLIGENCE AND ARTIFICIAL NEURAL NETWORK and submitted to the Department of Computer Science & Engineering at Center for Information Technology, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my work carried out during a period from **January, 2016** to **April, 2016** under the supervision of **Mr. Gunjan Chhabra** *Centre of Information Technology, UPES, Dehradun.*

The matter presented in this project has not been submitted by me for the award of any other degree of this or any other University.

Glen Bennet Hermon
Roll No.: R102214003
SAP ID: 500036672

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 12th April 2016

Gunjan Chhabra
(Project Guide)

Dr. Amit Agarwal
Program Head – M. Tech (AI & ANN)
Center for Information Technology
University of Petroleum & Energy Studies
Dehradun – 248 007 (Uttarakhand)

ACKNOWLEDGEMENT

I wish to express my deep gratitude to my guide **Mr. Gunjan Chhabra,** *Centre of Information Technology, UPES, Dehradun*, for all advice, encouragement and constant support they have given to me throughout my project work. This work would not have been possible without their support and valuable suggestions.

I am heartily thankful to my course coordinator, **Mr. Vishal Kaushik**, for the precise evaluation of the milestone activities during the project timeline and the qualitative and timely feedback towards the improvement of the project.

I sincerely thank to our respected **Dr. Amit Agarwal**, Program Head of the Department, for his great support in doing our project in **M. Tech Artificial Intelligence and Artificial Neural Networks** at **CIT**.

I am also grateful to **Dr. Manish Prateek, Associate Dean** and **Dr. Kamal Bansal** Dean CoES, UPES for giving me the necessary facilities to carry out my project work successfully.

I would like to thank all my **friends** for their help and constructive criticism during my project work. Finally, I have no words to express my sincere gratitude to my **parents** who have shown me this world and for every support they have given me.

Name **Glen Bennet Hermon**

Roll No. **R102214003**

SAP ID **500036672**

ABSTRACT

Large organizations and institutions maintain ledgers which is a permanent summary of all amounts entered in supporting journals which list individual transactions by date. A company's financial statements and balance sheets are generated from summary totals in the ledgers. Adjusting journal entries (and Ledger adjustments) are used by all companies that comply with generally accounting principles, and are used to adjust a company's revenue and expense accounts to ensure that all business activity has been included in the company's financial results, even if a cash exchange did not take place or the transaction wasn't processed through accounts payable and accounts receivable.

These account adjustments are currently handled manually by large workforces employed by large firms. Companies handling the Banking and Financial sectors with high stakes in the current market spend huge investments to maintain this workforce that do these adjustment procedures. These adjustments are very heuristic and is based on the decisions these users take hugely based on prior experience and that of a deep understanding of the relationships between attributes of such accounting procedures and adjustment techniques. Hence there is a proposed need of applying Machine Learning to historical data of past adjustment records to have a system that suggests the most probable adjustment given a scenario and possible adjustments based on similar adjustment patterns made in the past. The implementation of such a system will drastically reduce the cost and the requirement of huge investments for such workforces. This requirement was proposed by the institution I am interning at [L&T Infotech, Mumbai (Powai branch)], under the department that serves the requirements of another organization that deals with the Banking and Finance sector.

Keywords: *Adjustments, Ledger, Ledger adjustments, accounting procedures, Banking and Finance, heuristic, Machine Learning, historical data, reduce the cost.*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 History

When we think of how accounts were handled before the current digital era, we've got to bring back memories of all the hard printed copies and stacks of receipts and handwritten records, quite cumbersome to even think about it too, but those were the days where **journals** were notebooks in which a bookkeeper noted down entries after sales got closed, expenses were incurred, revenues received or any event as such occurred that had an altering effect on the company's accounts.

Nowadays, in the luxury of 'soft-files' handled by 'software', as a part of an accounting system, the concept of the journal is implemented automatically through point of sail systems or manually via onscreen forms. Since this is the software implementation of the concept of journals, error checking and user guidance are provided so that it is ensured that the correct accounts are impacted, and for the fact that the entries of credit and debit are properly registered. The advent of software in this field of accounting brings with it numerous advantages of automation such as automatically posting entries of a journal to a ledger, which is the second stage of the cycle of accounting.

If we look into the origins of the word 'journal', it has its roots in Old French and Latin, the meaning hints at daily activity as 'jour' is the French word for day. It is for the same reason that newspapers and personal diaries are sometimes called journals. Journals are usually updated continuously or at least on a daily basis, compared to other records which may be updated not as frequently. A running list of all account-impacting transactions are maintained in Journals as and when they occur. At any given point of time if a specific question is ever raised about transactions that occurred on a given day, the answer may be found in the journal.

1.1.1 Daybooks

There also exists another entity called the daybook which may be used in some organizations like a book that maintains original entry, like an original point of data entry for transactions. As it is the point of original data entry, the chronological order is maintained just as in journals. In the cycle of accounting the day book can be placed at the start of the cycle. This is usually taken into consideration when there is a need to put direct record capturing capabilities to the people engaged

in transaction activities. Such a practice is also taken into consideration when need arises to maintain additional transaction information, or maybe there is a desire to maintain specific kinds of entries together, and likewise for other entries such that there may exist separate books like, 'sales daybook' or even more specific like a sales daybook specific for each region or maybe a 'cash daybook' for keeping cash transactions together.

When we speak about daybooks we may also take in consideration a similar concept of what sub ledgers are in relation to ledgers analogous to daybooks in relation to journals. The structure of a daybook (likewise, a sub ledger) is similar to that of its parent, the journal (likewise, a ledger), the only difference may be that there will be transactions with additional details, with information on customers, venders, etc., that may not be carried on the parent record. But when the use of a daybook is considered, the entries are passed chronologically to the journal, in an appropriate form in the layout of credits and debits to the accounts that the organization holds.

1.2 Journal

As mentioned earlier a journal is used for recording transactions for accounting and bookkeeping as and when they occur. In the cycle of accounting, the first step is usually that of entering transactional data into a journal. This cycle of accounting follows the method of double entry where the entries for bookkeeping are recorded by crediting to one or more accounts while debiting from one or more accounts with the same amount in total.
The Journal is also called the book for original entries, since the time that before manually recording transactions to accounts in the general ledger or other sub ledger, the transactions were first entered in a journal.

1.3 Ledger

The main principle book, the file that holds records of all transactional and totaling entries for economic variables maintained by monetary unit account type with credits and debits separately in columns with a fashion of a value for the starting balance and with a value that denotes the ending balance for each account is called a Ledger.
It acts as a standardized summary like a permanent record to all the amounts entered in respective journals where all individual transactions are listed by date.

Every company that maintains ledgers have all their financial statements generated from ledgers as a summary of totals.

1.3.1 Types

Sales Ledger: Records all accounts that are deemed receivable. Transactions done by the customers to the company financially are stored in this ledger.

Purchase ledger: Records all expenditure for purchasing made by the company

- General ledger: There are mainly 5 account types:
    - Assets
    - Liabilities
    - Income
    - Expenses and
    - Equity

1.4 Requirement Analysis

Account adjustments are currently handled manually by large workforces employed by large firms. Companies handling the Banking and Financial sectors with high stakes in the current market spend huge investments to maintain this workforce that do these adjustment procedures. These adjustments are very heuristic and is based on the decisions these users take hugely based on prior experience and that of a deep understanding of the relationships between attributes of such accounting procedures and adjustment techniques. Hence there is a proposed need of applying Machine Learning to historical data of past adjustment records to have a system that suggests the most probable adjustment given a scenario and possible adjustments based on similar adjustment patterns made in the past.

1.5 Objectives

The Machine Learning Implementation:

- The **subjective decisions taken by the users to make adjustments** are what I have focused on, **to figure patterns related** to various attributes within the transactions in consideration.
- By **extracting these probable patterns, I will be able to automate** the process of adjustments via generating suggestions based on the adjustments done previously by the user.
- **I have focused on the adjustments for costs and quantities mismatch**

The Interface:

- A grid interface that sports a table of 'entries that require adjustments'.
- A field for suggested adjustments (Outcome via machine learning).
- Check boxes to select multiple entries to accept suggestions in one go.
- Drop down menu for every entry that contains:
    - An option to accept suggested changes
    - Lists of other adjustments that can be made (manual override)
    - An option to make no change (Do nothing)

Modes of the System:

- Training mode, Where the training dada is provided
- Test mode, where adjustment suggestions are generated by the system based on the provided training data.
- Output: The accepted changes are recorded and a new data file is created containing the automated changes and the manual overrides respectively. [a field specifies whether manual or automated is also included]

1.6 PERT Chart

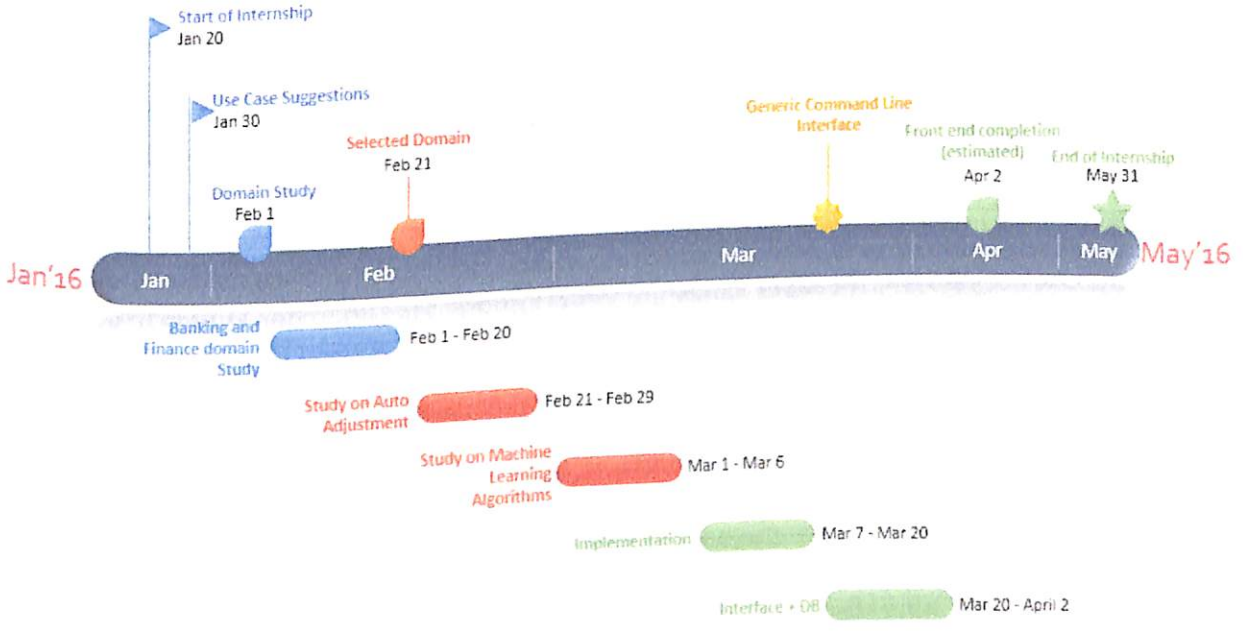


Fig 1.1 PERT Chart

2. SYSTEM ANALYSIS

2.1 Existing System

2.1.1 Maker-checker (The basic reconciliation Principle)

One of the fundamental principles of authorization for the information systems that govern financial organizations is the concept of Maker-checker (also known as 4-Eyes, or Maker and Checker). The mechanics of this maker checker principle hints to the necessary requirement of at least two individuals for its completion. When one individual creates a transaction the other individual confirms or authorizes that transaction in consideration. The actual importance lies in the segregation of these duties. With the functional division that of labor between classes of employees the control over data and system software is kept strict with this kind of a mechanism.

2.1.2 The Cycle of Accounting

Whenever business transactions are written into the journal, this marks the beginning of the cycle of accounting. Every company may have many accounts like 'cash on hand' or 'accounts receivable' or 'bank loans payable', talking about transactions in the cycle of accounting, they will cover all events that trigger an impact on these aforementioned accounts of the company. The chronological order-the sequence in accordance to occurrence-is how entries accumulate in the Journal. The step that comes next in the cycle of accounting is the step that involves the copying of entries in the Journal into a Ledger or multiple Ledgers, here in Ledgers, the organization is done by first organizing entries by account, and then entries within those accounts are organized chronologically.

To have an assurance that the right accounts on the basis of the convention of maintaining entries according to credits and debits are appropriately impacted, these software-based accounting systems usually have a proper error checks and user guidance in place [7]. Since software and automated processes go hand in hand, the job of copying journal entries to a ledger being the second step of the cycle of accounting is automated unlike the prior methods of manually moving the journal data entry by entry to the ledger one account at a time.

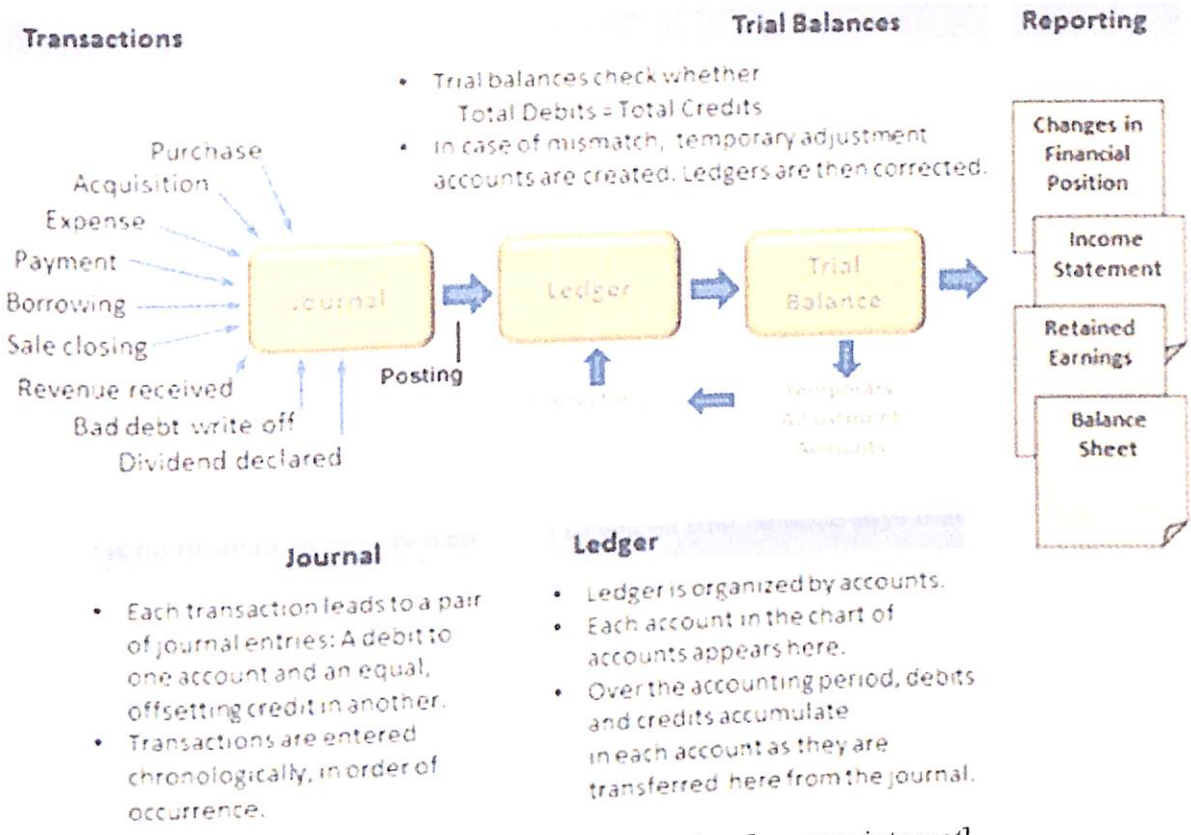


Fig 2.1 The Cycle of Accounting [*source: internet*]

2.1.3 Trial Balance

A basic rule of double-entry accounting is that for every credit there must be an equal debit amount. From this concept, one can say that the sum of all debits must equal the sum of all credits in the accounting system. If debits do not equal credits, then an error has been made. The trial balance is a tool for detecting such errors.

The double-entry rule of accounting has this mechanics in place that for each debit amount, there must be an equal credit amount. This concept further can be led to say that the total sum of credits must be equal to the total sum of debits in the accounting system. Hence, one can easily guess that if this sum of amounts does not match, then there is an error that's been made. The trial balance works like a tool for detecting the possibility of such errors.

The total sum of credits must be equal to the total sum of debits, if not, then there is a surety that an error was made in the process of accounting. Common errors that may occur are:

- Column totaling error, while totaling the columns.
- Error when transferring account balances while being copied to the respective trial balance

columns.

- Account omission by mistake.
- Account balance error.
- Error while a Journal entry is being posted.
- Error while transactions in a journal are recorded.

Working backwards from trial balance to the initial journal entry is the best and effective way for the isolation of an error

But there is no ultimate surety as well that a balanced trial balance says that there is no error. There is a chance that errors can be made by omission of entries wherein which a transaction wasn't recorded or a journal entry not posted, or posted to a wrong account in the ledger, or even a transpose of credit and debit entries.
Such possibilities of errors may not be found by the trial balance.

2.1.4 Adjusting Entries

In the accounting process, there may be economic events that do not immediately trigger the recording of the transaction [2].

These are addressed via adjusting entries which serve to match expenses to revenues in the accounting period in which they occur.

There are two general classes of adjustments:

- Accruals
- Deferrals

Accruals:

Revenues or expenses that have accrued but have not yet been recorded.
[Accrue: (verb) Charge that has been made, but not invoiced.]
So an adjusting entry is made to recognize the revenue in the period in which it was earned.

Deferrals:

Revenues or expenses that have been recorded but need to be deferred to a later date.
Meaning that the transaction is done prior to the period the transaction is meant for.
So a deferred entry (adjustment) is made to show the expense in the period in which it is in effect.

2.1.5 Closing Entries

The sequence of the closing process is as follows [3]:

1. Close the revenue accounts to Income Summary.
2. Close the expense accounts to Income Summary.
3. Close Income Summary to Retained Earnings.
4. Close Dividends to Retained Earnings.

2.1.6 Account Types

Every Journal/Ledger entry will impact at least two member accounts from this list [5]:

"Balance Sheet" account categories:

- 1. Asset accounts
- 2. Liability accounts
- 3. Equity accounts

"Income Statement" account categories:

- 4. Revenue accounts
- 5. Expense accounts

2.1.7 Credit and Debit

Every financial transaction brings a journal entry with **at least two equal and offsetting account changes**, the change in one account called a debit (DR) and the change in another account called a credit (CR).

When the journal entry is complete, the basic accounting equation holds and the Balance Sheet stays balanced:

Assets = Liabilities + Equities

And, for the account journal entries that follow from a single transaction:

Debits = Credits

Table 2.1 Credits and Debits[*source: internet*]

	Debit (DR) Entry ...	**Credit (CR) Entry ...**
Asset acct	Increases (adds to) account balance	Decreases (subtracts from) account balance
Liability acct	Decreases (subtracts from) account balance	Increases (adds to) account balance
Equity acct	Decreases (subtracts from) account balance	Increases (adds to) account balance
Revenue acct	Decreases (subtracts from) account balance	Increases (adds to) account balance
Expense acct	Increases (adds to) account balance	Decreases (subtracts from) account balance

2.1.8 Feeds (file for Adjustments or Transactions)

There is a daily entry of feeds which are files that contain all the **correction entries in a consolidated fashion.** These feed files are **received by the Reconciliation team** on a **daily** basis and is fed into the system. The system **automatically raises the mismatches.** These **errors are raised** in the system for correction.

2.1.9 Errors

The errors are basically of:

- **Total Cost mismatch or**
- **Total quantities mismatch**

The **User takes the decision of adjusting** these values based on **many subjective reasons.**

2.2 Motivations

After quite a few meetings and discussions based on the use-case feasibility and after analyzing the current market need, this requirement was proposed by the institution I am interning at [L&T Infotech, Mumbai (Powai branch)], under the department that serves the requirements of another organization that deals with the Banking and Finance sector.

The large companies that thrive in the banking and finance sectors of our current day industrial ecosystem spend huge investments on workforce that are assigned to manually perform all the transactional reconciliation and adjustments-in case of mismatches during reconciliation. These adjustments are very heuristic and is based on the decisions these users take hugely based on prior experience and that of a deep understanding of the relationships between attributes of such accounting procedures and adjustment techniques[10].

The motivation to provide a system that serves this specific sector by reducing these huge overheads of investment leads me to perceive a system that mimics a self-learning entity that can offer suggestions based on the past and historical records. Applying Machine Learning to historical data gives me a shot at a system that suggests the most probable adjustment given a scenario and possible adjustments based on similar adjustment patterns made in the past.

As technology is evolving, it is predicted that more and more business will shift to web as de facto platform. Web based nature of business or also referred as large scale business has many advantages which we can experience, however at the same time very few business ideas which use web as platform are able to survive.

Keeping these basics in mind I envision the final product to be that of a web based tool, basically a web app, to which one can upload a whole historical dataset containing all records of adjusted data based on errors and mismatches during reconciliation to the final commit from the source to the destination of any triggered transaction. Using which, the tool will create a decision model that will be used to provide suggestions to the most feasible adjustment to any new reconciliation mismatch that occurs.

The managers at the firm, the Banking and Finance sector at L&T Infotech (Powai), have also agreed that after the development of my suggested machine learning system as a POC (Proof of concept) they will look into integrating this with a running (live) banking accounting system in place after the standard cycle of testing and approvals that the firm follows.

2.3 Problem Statement

The proposed need of applying Machine Learning to historical data of past adjustment records in an accounting system to have a convenience that suggests the most probable adjustment given a scenario and possible adjustments based on similar adjustment patterns made in the past.

This thesis will further explain all the different functionalities and procedures that occur in an accounting system and also will show the pathway towards understanding the machine learning methodologies implemented and also how and why they were chosen specifically.

2.4 Proposed System

2.4.1 Outcome as a Proposed Final System:

- A grid interface that sports a table of 'entries that require adjustments'.
- A field for suggested adjustments (Outcome via machine learning).
- Check boxes to select multiple entries to accept suggestions in one go.
- Drop down menu for every entry that contains:
    - An option to accept suggested changes
    - Lists of other adjustments that can be made [manual override]
    - An option to make no change [Do nothing]

2.4.2 Outcome as a minimum requirement of the internship:

- The learning engine (command line) that can a load training set of data (for training) and can evaluate test data based on the provided training set. Basically The training backend.

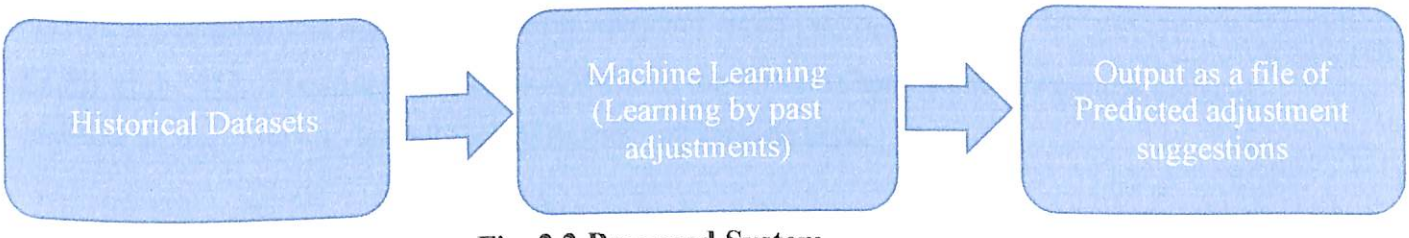


Fig. 2.2 Proposed System

2.4.3 Assumptions:

- Historical Adjusted Data (Test Data, having a good variation of occurrences) required as the initial training data for the system is provided.
- Data is in the form of flat files (like xls, xlsx or csv).
- Data file is manually uploaded (not integrated with the live system in place).

2.4.4 Modes of the System:

- Training mode, Where the training dada is provided
- Test mode, where adjustment suggestions are generated by the system based on the provided training data.
- Output: The accepted changes are recorded and a new data file is created containing the automated changes and the manual overrides respectively. [a field specifies whether manual or automated]

2.4.5 Major Time Consumers:

- The Training Backend (Based on Python)
- The Grid-UI front end (as developing a web app was the suggested implementation)

2.5 Modules

2.5.1 Data Collection and Consolidation

The basic understanding of the current system in place was a major task as to understand what type of data models are being generates in the system on a day to day basis.

The aspect to be worked on currently for me in the firm was **Indicative Data** which performs adjustments on **Loan and Deposit Products.**

The data provided would be that of **Common Balance** that is generated on a day to day basis.

A few terms specific to the company:

GENESIS: A consolidated server ecosystem where all the finalized data after reconciliation is stored. Basically a final data server base.

S2: A secondary server ecosystem similar to GENESIS. The data I would be provided was to be from this server base. The operations I had to carry out was on this system.

DELTA and FMA: These are third party systems in the firm that handled all the requirements of formatting all data into the desired form for executing specific tasks.

2.5.1.1 Basic levels of users with Finance and Risk Operations (FRO):

- Maker
- Checker

2.5.1.2 Basic functioning of the system (before Smart Adjustments):

- Excel files are uploaded from an adjustment portal to be approved.
- These adjustments are now categorized into:

1. Search and Retrieve:

- When a contract is already assigned (existing contract).
- Happens manually (as a Manual adjust entry).
- For Deposits.

2. Corrections:

- When there's no existing contract.
- Upload correction (adjustment).
- Create a new contract in genesis – The correction.
- Keys, such as GL (General Ledger) accounting keys, and many other keys are matched to find if it is an existing contract.

2.5.1.3 Possibilities of Adjustments [FRO-Finance and Risk Operations]:

Reconciliation Mismatches:

- GL (General Ledger) into genesis.
- Other Finance and Risk Services (FRS).
- Daily and Monthly GL.

The mismatches are the adjustments to be made.

Front-End Issues:

- Banking requests, etc.

Currency Exchange Fluctuation:

- Adjustments based on the differences between the **Base Amounts** and the **Functional Amounts.**

Month-End Adjustments:

- BD1 (Business Day 1) to BD8 (Adjustments after month end)

1. Exceptions for more business days may be approved.
2. GL adjustments can be made till the 15th of the next month.

- Done manually.

E.g.:

BD1: $100 adjustment uploaded

BD2: no change -> No Adjustment

:(might start processing from BD5

:

BD4: $500 adjustment

-> $500-$100 =>$400

-> $400 adjustment uploaded

:

:

BD8: The variance is calculated

likewise, and uploaded as a common balance file entry into GENESIS via the portal.

Synthetic Contract:

It is a Finance and Risk Service (FRS) reconciliation during the Month-End.

- The FRS record and Product Balances should match.
- Hence the variance (i.e., mismatches e.g.: $10) is made as a **synthetic contract** into GENESIS.
- Then the synthetic contract reconciliation is done.

2.5.2 The Machine Learning Backend

2.5.2.1 Two Processes:

1. BD1 to BD8 analysis.
2. **Manual** (Search and retrieve) and **Correction Entries.**

It is **Automated** by giving only the first file into the automation and then the common balance file is generated.

The files are generated from **DELTA** and **FMA** which contain:

- Contract Details.
- Product Details.
- Amounts.

2.5.2.2 Current status:

The automation is being worked on for **Products and Loans** and the current development is for the domain of **Retail** under that of **Positions** which is out of scope.

2.5.2.3 The Proposed Machine Learning Implementation

- The **subjective decisions taken by the users to make adjustments** are what I want to focus on, **to figure patterns related** to various attributes within the transactions in consideration.
- By **extracting these probable patterns, I will be able to automate** the process of adjustments based on the adjustments done previously by the user
- **I will be focusing on the adjustments for costs and quantities mismatch**

2.5.2.4 The Machine Learning Modules

The Look-ups on the modules I intended to be working on concerning machine learning are:

Estimators: which considers rules for calculating an estimate of a given quantity based on observed data: thus the rule (the estimator), the quantity of interest (the estimand) and its result (the estimate) are distinguished.

Decision Trees: They are a non-parametric supervised learning method used for classification and regression. The goal here is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Stochastic Gradient Descent: This is a simple yet very efficient approach to discriminative learning of linear classifiers under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

The advantages of Stochastic Gradient Descent are:

- Efficiency.
- Ease of implementation (lots of opportunities for code tuning).

2.5.3 The Web App Interface

- A grid interface that sports a table of 'entries that require adjustments'.
- A field for suggested adjustments (Outcome via machine learning).
- Check boxes to select multiple entries to accept suggestions in one go.
- Drop down menu for every entry that contains:
  - An option to accept suggested changes
  - Lists of other adjustments that can be made (manual override)
  - An option to make no change (Do nothing)
- Training mode, Where the training dada is provided
- Test mode, where adjustment suggestions are generated by the system based on the provided training data.
- Output: The accepted changes are recorded and a new data file is created containing the automated changes and the manual overrides respectively. (a field specifies whether manual or automated is also included)

3. DESIGN

The design of the proposed system is subdivided into the major modules of **Interface Flow**, **Data Flow** and the choice and implementation of the **Underlying algorithms**. The division of the system into the following categories gives a more efficient perspective on the software development life cycle for anyone that may work for the system, keeping the future in mind.

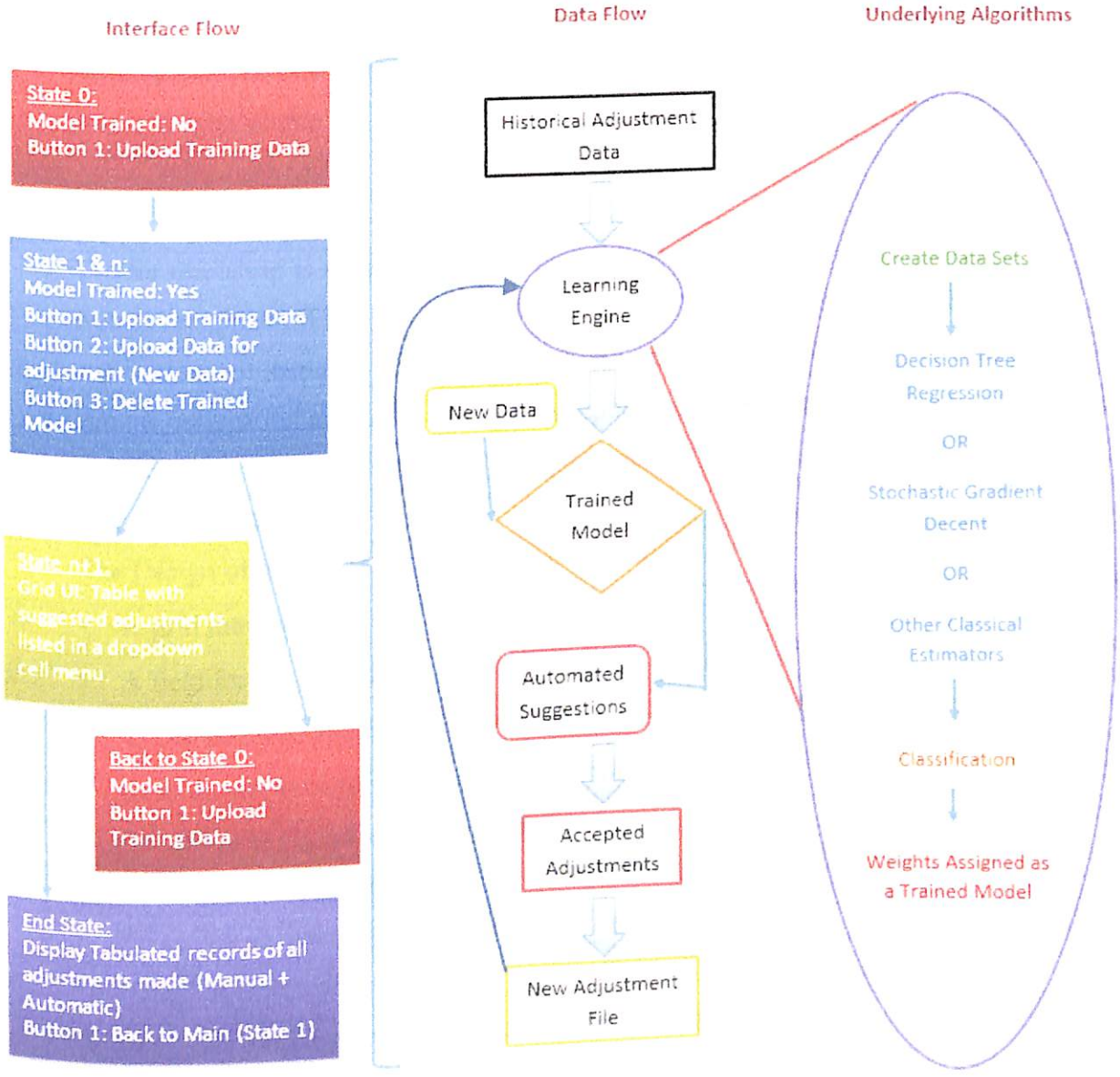


Fig 3.1 System Architecture and Design

3.1 Interface Flow

The web app is to be created in a Python-Django web framework environment over which the web UI elements will be furnished with the Twitter Bootstrap for the looks and a little bit of javascript for a good dynamic feedback

3.1.1 Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so one can focus on writing your app without needing to reinvent the wheel. It's free and open source.

3.1.2 Bootstrap

Twitter Bootstrap is a sleek, intuitive, and powerful front-end framework for faster and easier web development. It is a free and open-source collection of tools for creating websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. It aims to ease the development of dynamic websites and web applications.

3.1.3 Look and Feel

The Design of the interface as planned should contain

- A grid interface that sports a table of 'entries that require adjustments'.
- A field for suggested adjustments (Outcome via machine learning).
- Check boxes to select multiple entries to accept suggestions in one go.
- Drop down menu for every entry that contains:
  - An option to accept suggested changes
  - Lists of other adjustments that can be made (manual override)
  - An option to make no change (Do nothing)

3.1.4 The Flow of events

- **State 0**: The foremost and initial state, where the training dada is provided to the system that is ready to process new data
- **State 1&n:** This state has the Trained model in place from the Historical data provided in the previous state, here adjustment suggestions are generated by the system based on the provided training data. The user must accept or reject every suggested adjustment to proceed.
- **Final State (Output):** The accepted changes are recorded and a new data file is created containing the automated changes and the manual overrides respectively. (a field specifies

whether manual or automated is also included)

3.2 Data Flow

3.2.1 Procurement of data

All the historical reconciliation data that needs to be fed as the training data has to be consolidated into datasets to upload into the system. This initial data must contain all different variations in the adjustment choices that are possible in different scenarios as these will affect the future suggestions made by the system after it has been trained

3.2.2 Uploading as a file to the learning engine

The file has to be uploaded into the running web app which takes this uploaded file and passes it to the underlying python code that is the machine learning engine that generates the training model [14]. The training model can be made persistent if in case this will be implemented on a live system in the future

3.2.3 Adjustments and Final Output

After the system has generated the training model, the testing data or the file that requires suggestions on the adjustments that has to be made on the mismatches that have been raised by the reconciliation system can be uploaded into the system to generate suggested changes. The user has the choice to accept or reject individual adjustment suggestions or to accept all or reject all as a whole batch operation.

The final output is a new data file is containing the automated changes and the manual overrides respectively. A separate field in the file will specify whether the adjustments made were manual or automated.

3.3 Choice and Implementation of Underlying Algorithms

The underlying algorithms and functioning have been programmed in Python and considerable support has been taken from the Python package SciKit-Learn for the implementations and testing for the choice of the best machine learning algorithm suited for the problem in hand.

3.3.1 SciKit-Learn

Scikit-learn (formerly scikits.learn) is an open source machine learning library for the Python programming language.[2] It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-learn is largely written in Python, with some core algorithms written in Cython to achieve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

3.3.2 Decision Trees

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

3.3.2.1 Some advantages of decision trees are:

- Simple to understand and to interpret. Trees can be visualised.
- Requires little data preparation. Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data. Other techniques are usually specialised in analysing datasets that have only one type of variable.
- Able to handle multi-output problems.
- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
- Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

3.3.2.2 The disadvantages of decision trees include:

- Decision-tree learners can create over-complex trees that do not generalise the data well. This is called overfitting. Mechanisms such as pruning (not currently supported), setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree

are necessary to avoid this problem.

- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
- There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

3.3.2.3 Mathematical Formulation

Given training vectors $x_i \in R^n$, i=1..., l and a label vector $y \in R^l$, a decision tree recursively partitions the space such that the samples with the same labels are grouped together.

Let the data at node m be represented by Q. For each candidate split $\theta = (j, t_m)$ consisting of a feature j and threshold t_m, partition the data into $Q_{left}(\theta)$ and $Q_{right}(\theta)$ subsets

$$Q_{left}(\theta) = (x, y)|x_j <= t_m$$
$$Q_{right}(\theta) = Q \setminus Q_{left}(\theta) \quad \text{[source: internet]} \tag{1}$$

The impurity at m is computed using an impurity function $H()$, the choice of which depends on the task being solved (classification or regression)

$$G(Q, \theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta)) \quad \text{[source: internet]} \tag{2}$$

Select the parameters that minimises the impurity

$$\theta^* = \text{argmin}_\theta \, G(Q, \theta) \text{[source: internet]} \tag{3}$$

Recurse for subsets $Q_{left}(\theta^*)$ and $Q_{right}(\theta^*)$ until the maximum allowable depth is reached, $N_m < \min_{samples}$ or $N_m = 1$.

3.3.2.4 Complexity

In general, the run time cost to construct a balanced binary tree is $O(n_{samples} n_{features} \log(n_{samples}))$ and query time $O(\log(n_{samples}))$. Although the tree construction algorithm attempts to generate balanced trees, they will not always be balanced.

3.3.3 Stochastic Gradient Decent

Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to discriminative learning of linear classifiers under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

SGD has been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing. Given that the data is sparse, the classifiers in this module easily scale to problems with more than 10^5 training examples and more than 10^5 features.

3.3.3.1 The advantages of Stochastic Gradient Descent are:

- Efficiency.
- Ease of implementation (lots of opportunities for code tuning).

3.3.3.2 The disadvantages of Stochastic Gradient Descent include:

- SGD requires a number of hyperparameters such as the regularization parameter and the number of iterations.
- SGD is sensitive to feature scaling.

3.3.3.3 Mathematical Formulation

Given a set of training examples:
$(x_1, y_1), \ldots, (x_n, y_n)$where $x_i \in \mathbf{R}^n$ and $y_i \in \{-1, 1\}$,[*source: internet*] (4)
our goal is to learn a linear scoring function $f(x) = w^T x + b$ with model parameters $w \in \mathbf{R}^m$ and intercept $b \in \mathbf{R}$. In order to make predictions, we simply look at the sign of $f(x)$. A common choice to find the model parameters is by minimizing the regularized training error given by

$$E(w,b) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i)) + \alpha R(w) \quad \text{[source: internet]} \tag{5}$$

where L is a loss function that measures model (mis)fit and R is a regularization term (aka penalty) that penalizes model complexity; $\alpha > 0$ is a non-negative hyper parameter.

Stochastic gradient descent is an optimization method for unconstrained optimization problems. In contrast to (batch) gradient descent, SGD approximates the true gradient of $E(w,b)$ by considering a single training example at a time.

The class **SGDClassifier** implements a first-order SGD learning routine. The algorithm iterates over the training examples and for each example updates the model parameters according to the update rule given by

$$w \leftarrow w - \eta(\alpha\frac{\partial R(w)}{\partial w} + \frac{\partial L(w^T x_i + b, y_i)}{\partial w}) \quad \text{[source: internet]} \tag{6}$$

where η is the learning rate which controls the step-size in the parameter spaces. The intercept b is updated similarly but without regularization.
The learning rate η can be either constant or gradually decaying. For classification, the default learning rate schedule (learning rate='optimal') is given by

$$\eta^{(t)} = \frac{1}{\alpha(t_0 + t)} \quad \text{[source: internet]} \tag{7}$$

where t is the time step (there are a total of n_samples * n_iter time steps), t_0 is determined based on a heuristic.

3.3.3.4 Complexity

The major advantage of SGD is its efficiency, which is basically linear in the number of training examples. If X is a matrix of size (n, p) training has a cost of $O(kn\bar{p})$, where k is the number of iterations (epochs) and $\bar{p}$ is the average number of non-zero attributes per sample.

3.3.4 Choice of Algorithm

The choice of algorithm was based on the ease of implementation as per the data that was being generated by the system.
Data for reconciliation comes from the checker from the maker checker cycle to be matched and reconciled. The mismatches are raised as separate error alerts and passed as separate files called FEEDS back to the system. On a daily basis there is approximately an average of 500 to

700 feeds that are passed to be adjusted.

For the efficiency of implementation, the data of these feeds are consolidated on to a single file that acts as a batch-wise dataset.

Since the implementation of batch-wise learning and classification was more feasible in the form of mini-batches in the SGD algorithm (Stochastic Gradient Decent), the final choice was made to go ahead with the SGD algorithm, after an extensive effort on trying to learn methods for persistent implementations analogous to the mini-batches for the Decision Tree algorithm.

4. IMPLEMENTATION OF SMART ADJUSTMENTS

The system implemented thus far is capable of taking inputs as csv files as a training set as well as texting (evaluation) datasets for the SGD (Stochastic Gradient Decent) estimator which then makes the training model as a partial modal so as to avail modifiability to mold to new renditions of the training model based on new data being fed into the system.

4.1 Sample Reconciliation Data

Key_ID|City|Spend_Cap|Transaction_Freq

0000|Pune|Basic|6

0001|Pune|Basic|6

0008|Pune|Mod|10

0012|Pune|High|5

0013|Pune|High|7

0016|Pune|High|4

0017|Pune|High|8

0018|Pune|High|6

0019|Pune|High|5

0020|Pune|Basic|6

4.2 Sample General Adjustment Data

Generalledger_ID|Maker|Adjustment_Value|Checker|Medium|Value|Previous_Sale|Similar_Acc_Key

0000|JRRT|500|Harpercollins| Physical|4.5|150000000|0002,0003,0001,0005,0004

0001| ChaD|250|Penguin| Physical|4.6|200000000|0004,0005,0000,0002,0003

0002| JRRT|250|HarperCollins| Physical|4.5|146000000|0000,0003,0001,0005,0004

0003| DanB|200|RHUK| Physical|4.8|80000000|0000,0002,0001,0005,0004

0004| CSLe|200|UKChildrens| Physical|4.6|85000000|0000,0001,0002,0003,0005

0030| TarD|250|VakilsFefferSimons| Physical|5.0|10000|0031,0032

0031| JeoS|150|JaicoPublishingHouse| Physical|3.9|1000|0030,0032

0032| MamM|150|MalayalamManorama| Physical|4.2|100000|0030,0031

0033| WalI|450|Little| Physical|4.7|250000|0034

0034| MohG|175|Rajpal| Physical|4.2|100000|0033

4.3 Sample Transaction Data

Transaction_ID|Month|Acc_ID|General ledger_ID

0000|Jan|0000|0032

0001|Jan|0000|0021,0022

0002|May|0000|0006,0007

0003|May|0000|0033,0034

0004|Oct|0000|0023,0024

0005|Oct|0000|0004

0006|Jan|0001|0000

0007|Jan|0001|0021,0022

0008|May|0001|0006,0031

4.4 About Data Fields

From the data one can see that there are three entities under analysis.

- Ledger Keys (Work as accounting keys made to perform transacts specific to the account in consideration)
- Products
- Transactions

Properties of these three entities which are under analysis are as follows.

For judging the proper account on which the adjustments for reconciliation have to be made, I will be looking at:

- Account Key ID: which will act as primary key for the dataset.
- City: that tells more about how easily large scale is accessible.
- Spend capacity: how much revenue that customer can generate
- Transaction frequency: how often the transactions will happen
- Platform: may be used to push the offers

For General Adjustments I will be looking at:

- General ledger ID: which will act as the primary key for the dataset.
- Maker: who wrote the general ledger that might influence the transact.
- Adjustment Value: decides worth of the general ledgers to large extent
- Checker: may influence the reconciliation mismatch of general ledger
- Medium: decides portability of the general ledger
- Value: another influencing factor to buy a general ledger
- Previous sale: tells more about the quality aspect of product
- Similar accounting Key: will help the system to suggest other probable adjustments

For transactions I will be looking at,

- Transaction ID: Which will act as primary key of the dataset
- Month: When the transaction happened
- Transact Account ID: The related Account in which the transaction occurred.
- General ledger ID: The current working Ledger.

This file will be the basic one before heading into the adjustments section that looks into the reconciliation mismatches.

4.5 About the Algorithm Flow

Algorithms to find the similarity measure. The algorithm is as mentioned below.

Input to the algorithm:

- Set of pre classified customer tuples 'X'
- Set of known class labels 'Y'
- New unclassified customer tuple 'Z'
- K value

Algorithm steps:

- For all Xi from set X
    - Calculate the distance Di using weight and importance of attribute value and attribute between Xi and Z (Euclidean dist.)
- Find maximum K distances. That is all those tuples which are very similar to the new tuple Z
- Assign the label from known set of labels 'Y' to new tuple Z which occurs most number of times in selected K most similar tuples.

This algorithm is fairly simple given the importance and weight factors are assigned correctly. This will classify the new adjustment when it gets registered as a new choice. This will give the model a fair idea about what kind of choices the user makes based on the cost or quantities mismatch. Classification is important as this tells the model about the adjustment probability for new adjustments to made for the first time. In short it will help the model to identify new potential adjustment patterns.

Challenges with this technique is that being an instance based classification technique, it will work only when large has datasets about previous reconciliation and the adjustments are provided. But since it is meant to be integrated with a live functioning system which has databases maintained it provides feasible as the required data is in place.

Second challenge is that of cleanliness of the data available. The collected data can have outliers and here in this work it is expected that the data is cleaned using data cleansing techniques. Also many times this data can have wrong or misleading information. That might change the output of the algorithm and the give me misclassified adjustment tuple.

5. LIMITATION AND FUTURE ENHANCEMENTS

5.1 Limitations

Large scale companies, specifically service based companies face many issues due to overhead of management of huge transactional adjustment calls, investment that goes in maintaining this system, lack of understanding of these heuristically taken decisions based on many factors that need experience, multi linked attributes and reasons that may not be retrieved from the values stored in the datasets, heavy competition from over-priced unstable solutions. All these issues form three sub problems of this bigger problem, they are,

- Lack of understanding of Major Heuristics.
- Lack of resources – both financial and that of data.
- Lack of structures linkages with data that actually belong to each other.

5.2 Directions for Future Work

Changes and improvements are possible with this model. In cases where there is need of classification, I can add different classification techniques such as decision tree classifiers and naïve Bayes classifier along with K Nearest Neighbors so that the classification can be more robust. Even addition of neural networks will be beneficial. Along with hierarchical techniques I can add support Vector Machines so that when the geometry of the data is non convex, the clustering can be more accurate.

When the dataset grows in terms of number of records, the process becomes very slow. To avoid such a scenario, I can also add support for sampling techniques which will draw a sample that is of smaller size yet maintains all the characteristics of the dataset. I can use algorithms (like: Chernoff bounds) to decide the optimal sample size.

Such improvements and integration of this model with the actual working business scenario will be the future roadmap for this proof of concept work.

The finalized implementation will be like a backend server code that keeps checking and correcting mismatches in accounting ledgers and transactions in a very heuristic fashion with a high percentage of accuracy so that the investments currently to do this work will be totally reduced.

6. SCREENSHOTS

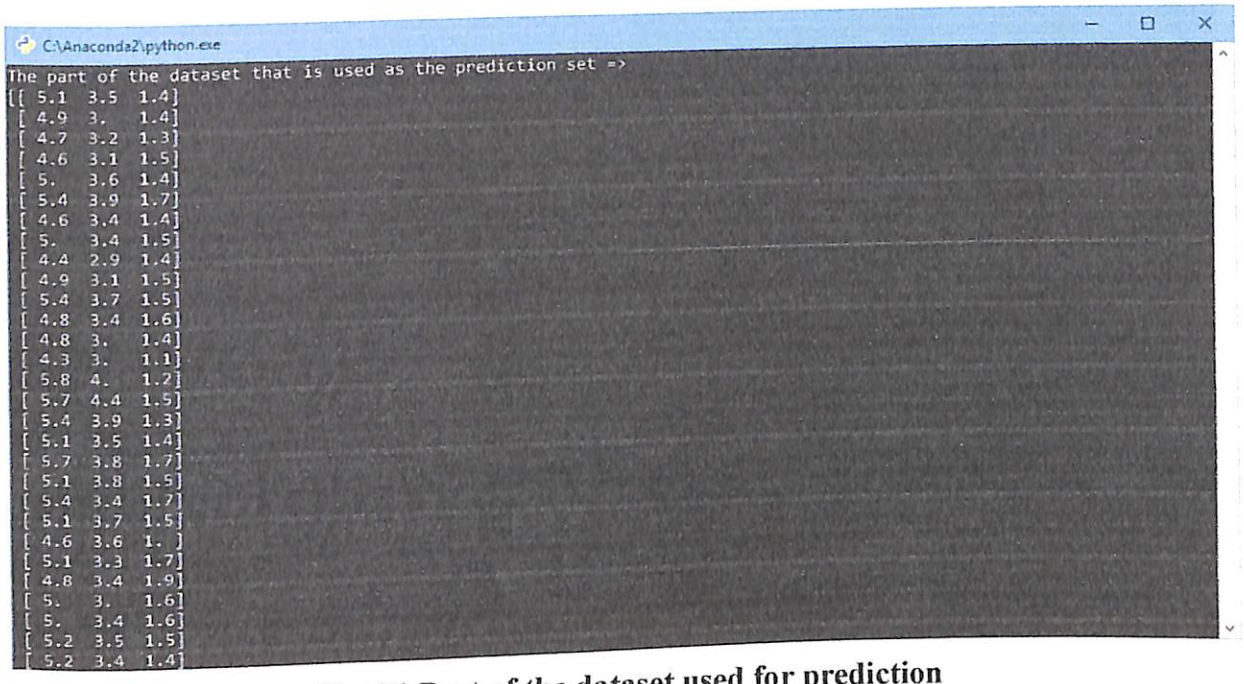


Fig 6.1 Part of the dataset used for prediction

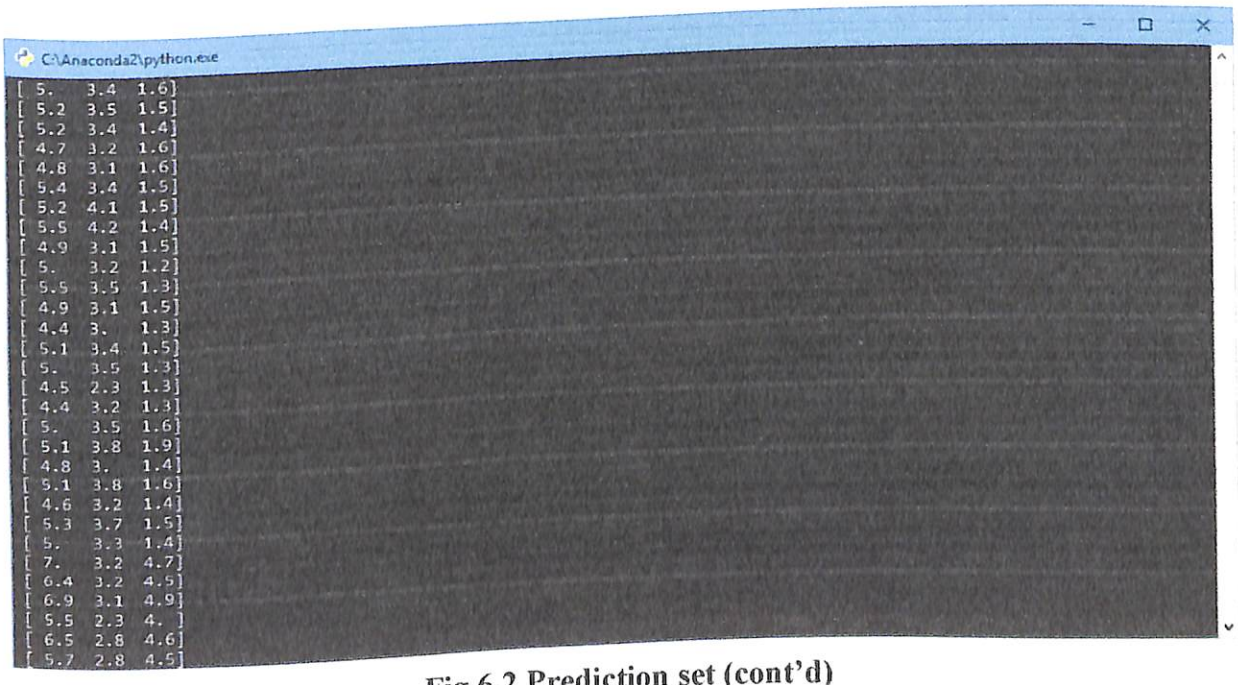


Fig 6.2 Prediction set (cont'd)

Fig 6.3 Prediction set (cont'd)

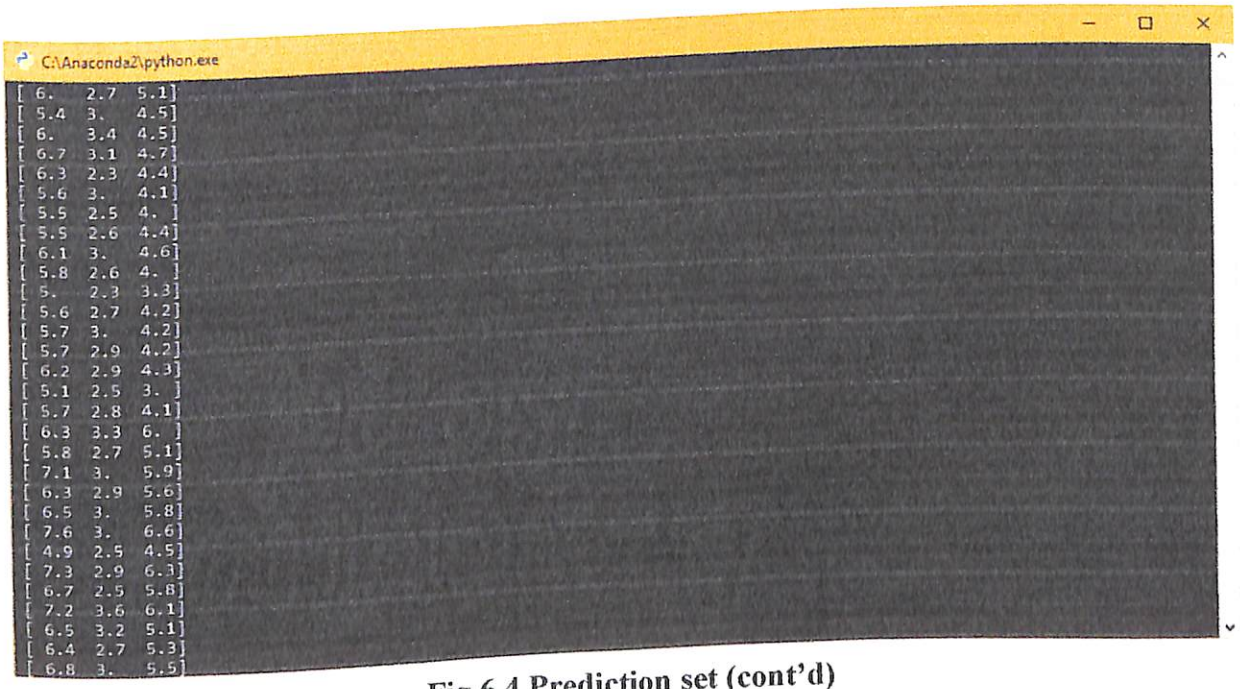


Fig 6.4 Prediction set (cont'd)

Fig 6.5 Prediction set (cont'd)

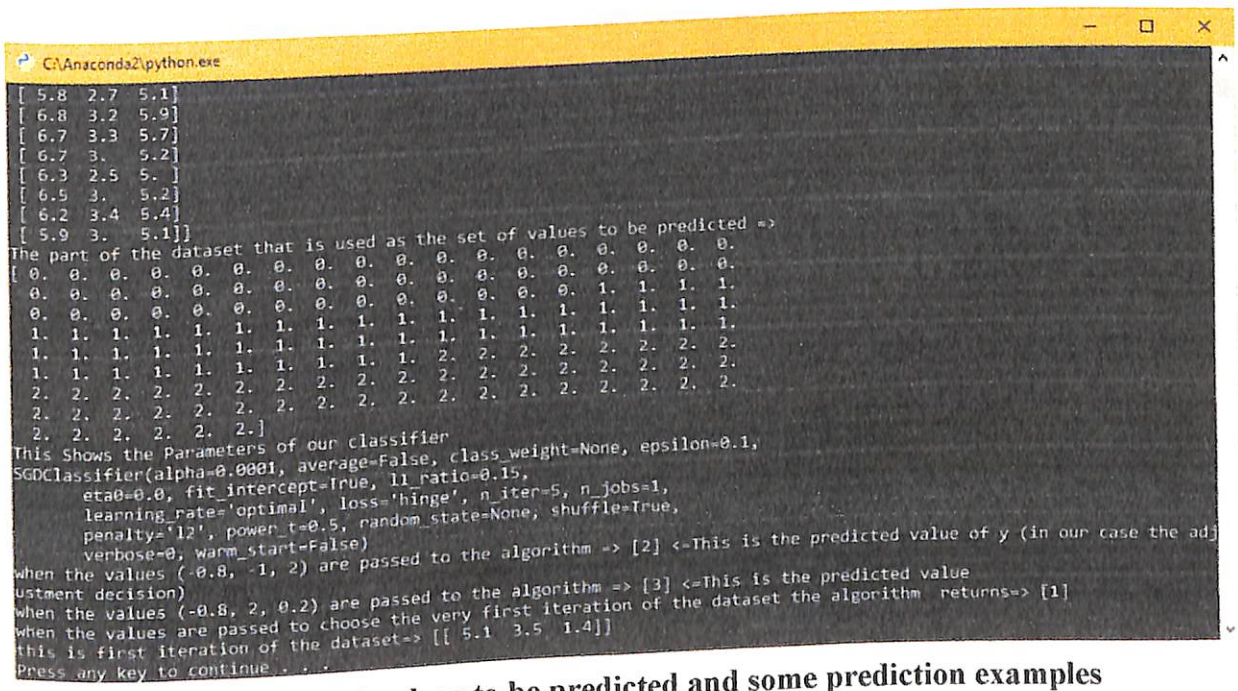


Fig 6.6 Related values to be predicted and some prediction examples

7. CONCLUSION

This project thesis forms a proof of concept to demonstrate the power of Machine Learning techniques in the area of Ledger Adjustments during Reconciliation mismatches for those firms that deal with the banking and finance sectors in the current economic ecosystem. The verification of this project will need integration of this model with actual business scenario. It is hardly possible to integrate the same with working business scenario so I have verified the implementation using the SGD (stochastic gradient decent). I used only sample data to prove the functioning of the proof of concept. I found that the various data mining techniques implemented to create and get this model in working condition have given me expected results. I have no other way apart from manual verification to verify the result at this stage.

REFERENCES

[1] Jiawei Han and Micheline Kamber, Data Mining Concepts and Techniques, San Francisco, Morgan Kaufmann Publishers, Elsevier, 2011.

[2] James Don Edwards, PhD, D.H.C., Roger H. Hermanson, PhD., Accounting Principles: A Business Perspective, First Global Text Edition, Financial Accounting [Volume 1, Issue 3, October 2012].

[3] JOURNAL ENTRY Reference Manual [www1.umn.edu/ohr/prod/groups/ohr/@pub/@ohr/@trainingservices/documents/asset/ohr_asset_084471.pdf].

[4] http://www.cuanswers.com/pdf/cb_ref/G-GLBalancingtools.pdf

[5] http://scikit-learn.org/stable/documentation.html

[6] www.business-case-analysis.com

[7] Tom Mitchell, Machine Learning, New York, McGraw Hill Education, 2013.

[8] Stuart Russell and Peter Norvig, Artificial Intelligence A Modern Approach, New Delhi, Pearson Education, Dorling Kindersley Publishing, India Edition, 2011.

[9] Mark Lutz, Learning Python, O'reilly Publishing, 2013.

[10] Amazon General ledgers Dataset, http://authorearnings.com/report/the-50k- report/

[11] Amazon Product Dataset, https://snap.stanford.edu/data/amazon- meta.html

[12] Customer Information, http://www.slideshare.net/mitraarnab1/india- on-internet-2014

[13] Customer Information, http://www.businessinsider.in/The-Surprising- Demographics-Of-Who-Shops-Online-And-On- Mobile/articleshow/36449798.cms

[14] E.W.T. Ngai, Li Xiu and D.C.K. Chau, "Application of Data Mining Techniques in Customer Relationship Management : A Literature Review and Classification", Expert Systems With Applications, 2009.

[15] R. Ling and D.C. Yen, "Customer relationship management : An Analysis framework and implementation strategies", Journal of computer information systems, 2001.

[16] E.W.T. Ngai, "Customer relationship management research (1992-2002): An

academic literature review and classification", Marketing intelligence, planning, 2005.

[17] Ruchi Nayyar and Dr. L. S. Gupta, "Impact of Changing Demographic Profiles of Indian Customers on Their Internet Shopping Behavior", ViewPoint, 2010.

[18] Dahiya Richa, "Impact of Demographic Factors of Consumers on Online Shoping Behavior: A Study of Consumers in India", International Journal of Enginerring and Management Sciences, Vol 3(1), 2012, 43-52.

[19] Armando Roggio, Practical Large scale, Editor, http://www.practicalecommerce.com/articles/75484-8-Reasons-Why-Ecommerce-Businesses-Fail

[20] David Rekuc, Marking Land, Columist, http://marketingland.com/brands-fail-large scale-fix-122172