Prediction of lost customers in the telecommunications industry utilizing machine learning on large data platforms

Sasidhar K, Dr. Rajesh Kulkarni
Research Scholar Career Point University KOTA, Rajasthan.
Research Supervisor Career Point University KOTA, Rajasthan.

Abstract:
The telecom industry study is crucial for boosting businesses' profitability, particularly by accurately predicting churn. This research focused on developing a customized churn prediction system for SyriaTel, a telecom company. High AUC values were essential for precise churn forecasts, and the dataset was split into 70% training and 30% testing sets. Cross-validation aided in reliable model assessment and hyper parameter tuning. Feature engineering and selection techniques were employed to prepare the features for machine learning algorithms. Addressing data imbalance, under-sampling and tree-based algorithms were utilized. Four tree-based models were chosen: Decision Tree, Random Forest, Gradient Boosting Machine, and XGBOOST. Success relied on strategic planning and inclusion of mobile social network features. XGBOOST outperformed with a 93.301% AUC on the SyriaTel dataset, followed by GBM, Random Forest, and Decision Tree. Testing with a new dataset showed XGBOOST's AUC at 89%. Regular model retraining is necessary due to non-stationary data. Incorporating Social Network Analysis improved churn prediction in telecom

Keywords: boosting, feature engineering, selection techniques, churn, dataset.
DOI: 10.24297/j.cims.2023.6.24

1. Introduction

Consumer behavior analytics is a field that aims to understand and predict consumer behavior based on data analysis. Fuzzy rule-based classification, data mapping, bagging, and MapReduce are all techniques used in this context. Let's break down each component and see how they come together.

Fuzzy Rule-Based Classification:

Fuzzy logic allows for handling uncertainty and imprecision in data by assigning membership degrees to different categories. Fuzzy rule-based classification uses a set of fuzzy if-then rules to classify data instances. Each rule consists of antecedents (conditions) and consequents (actions).

Fuzzy rule-based systems are effective for dealing with complex and ambiguous data sets, making them suitable for consumer behavior analysis.

Data Mapping:
Data mapping involves transforming and aligning data from different sources or formats to a common representation. In consumer behavior analytics, data may come from various sources like social media, online transactions, or surveys. Data mapping ensures that the data is in a consistent and usable format for analysis. It may involve standardizing attributes, resolving inconsistencies, or merging data from multiple sources.

Bagging:
Bagging, short for bootstrap aggregating, is an ensemble learning technique where multiple models are trained on subsets of the data, and their predictions are combined to make a final prediction. In consumer behavior analytics, bagging can be used to improve the accuracy and robustness of classification models. By training multiple fuzzy rule-based classifiers on different subsets of consumer behavior data, bagging reduces the risk of overfitting and provides more reliable predictions.

MapReduce Scheme:
MapReduce is a programming model and framework for processing large-scale data in a distributed computing environment. It divides the data processing into two stages: map and reduce. The map stage involves breaking down the data into smaller chunks and performing initial processing on each chunk. The reduce stage involves aggregating and summarizing the results from the map stage. MapReduce is particularly useful for handling big data analytics tasks efficiently.

In the context of consumer behavior analytics, a MapReduce scheme can be employed to process large volumes of consumer data. The mapping phase can involve data preprocessing steps like data cleaning, transformation, and mapping. The fuzzy rule-based classification and bagging processes can be parallelized and distributed across multiple nodes in a MapReduce cluster. This allows for scalable and efficient analysis of consumer behavior data. By combining fuzzy rule-based classification, data mapping, bagging, and MapReduce, consumer behavior analytics can benefit from improved accuracy, handling of uncertainty, scalability, and efficiency in processing large volumes of data. These techniques enable organizations to gain valuable insights into consumer behavior, make informed decisions, and develop targeted marketing strategies.

1.1 Fuzzy Rule-Based Classification

Fuzzy rule-based classification is a technique used in machine learning and data analysis to categorize data instances into different classes based on fuzzy logic and a set of predefined rules. It extends traditional crisp (binary) rule-based classification by allowing for the representation of uncertainty and imprecision in data. In traditional crisp rule-based classification, rules are defined using crisp (exact) conditions and actions. For example, a rule might state, "If the age is greater than 30 and the income is above $50,000, then classify the individual as a high-spending customer."

In fuzzy rule-based classification, fuzzy logic is employed to handle the uncertainty and imprecision in data. Fuzzy logic allows for assigning membership degrees to different categories rather than a strict binary classification. This means that an instance can partially belong to multiple classes with varying degrees of membership.

Fuzzy rule-based classification consists of two main components:

Fuzzification: In this step, the input data is transformed into fuzzy sets by assigning membership degrees to relevant categories. Membership functions define the degree to which an instance belongs to a particular category. For example, an age of 35 might have a membership degree of 0.7 for the category "young" and 0.3 for the category "middle-aged."

Rule Evaluation and Aggregation: In this step, fuzzy rules are evaluated based on the membership degrees of the input data. Each rule consists of antecedents (conditions) and consequents (actions). The antecedents are evaluated based on the membership degrees of the input data, and the consequents provide the output or classification decision. Multiple rules may contribute to the classification decision, and their outputs are aggregated using fuzzy logic operations like the max, min, or average.

The classification decision is made based on the aggregated outputs of the fuzzy rules. The output can either be a crisp class label or a fuzzy set representing the degree of membership in different classes.

Fuzzy rule-based classification is particularly useful in domains where there is uncertainty, ambiguity, or imprecision in data. It allows for more flexible and nuanced classification decisions compared to traditional crisp rule-based approaches. This technique has been successfully applied in various fields, including consumer behavior analysis, pattern recognition, decision support systems, and expert systems.

1.2 Data Mapping

Data mapping is the process of transforming and aligning data from one format or structure to another. It involves mapping the attributes or fields of data from a source to a target representation. Data mapping is commonly used in data integration, data migration, and data transformation tasks to ensure that data is properly understood and utilized in the target system or application.

Here are the key steps involved in data mapping:
Source and Target Definition: Identify the source data, which could be in a database, file, API, or any other format. Similarly, define the target data format, considering the structure and requirements of the destination system or application.

Attribute/Field Mapping: Examine the attributes or fields present in the source data and map them to their corresponding counterparts in the target data. This involves identifying which attributes align with each other and determining how the data should be transformed or manipulated during the mapping process. It may also involve handling missing or incompatible attributes.

Data Transformation: If necessary, apply data transformation operations during the mapping process. This can include data type conversions, calculations, aggregations, filtering, or any other manipulations required to match the target data format or business rules.

Handling Complex Mapping Scenarios: In some cases, data mapping may involve more complex scenarios, such as handling hierarchical data structures, mapping data across multiple sources, or dealing with data that requires normalization or denormalization. These scenarios require careful analysis and mapping strategies to ensure accurate and meaningful data integration.

Validation and Testing: After mapping the data, it is crucial to validate and test the mapping to ensure that the transformed data aligns with the target requirements and expectations. This involves verifying the accuracy and integrity of the mapped data through various testing techniques like sample data comparison, data profiling, or automated validation scripts.

Documentation: Document the data mapping process, including the source and target definitions, attribute mapping rules, transformation operations, and any assumptions or considerations made during the mapping process. This documentation serves as a reference for future use and facilitates understanding and collaboration among stakeholders.

Data mapping plays a vital role in data integration and data management processes. It ensures that data from diverse sources can be effectively and accurately utilized in different systems, applications, or analytical processes. By mapping data appropriately, organizations can achieve data consistency, interoperability, and seamless data flow across various systems and platforms.

1.3 Bagging

Bagging, short for Bootstrap Aggregating, is a machine learning ensemble method used to improve the performance and robustness of predictive models. It involves creating multiple subsets of the original training dataset through a process called bootstrapping and training a separate model on each subset. These individual models are often referred to as base learners.

The bagging algorithm works as follows:

Bootstrap Sampling: Randomly sample the training dataset with replacement to create multiple bootstrap samples. Each bootstrap sample is the same size as the original dataset but may contain duplicate instances and exclude some original instances.

Model Training: Train a separate base learner on each bootstrap sample. The base learner can be any machine learning algorithm, such as decision trees, support vector machines, or neural networks.

Ensemble Aggregation: Combine the predictions from all the base learners to make a final prediction. The aggregation method varies depending on the problem type. For regression tasks, the predictions are often averaged, while for classification tasks, voting or averaging probabilities is commonly used.

Bagging offers several advantages:

Reduced Variance: By training multiple models on different bootstrap samples, bagging reduces the variance of the final prediction. It helps to stabilize the model by reducing the impact of outliers or noisy instances.

Improved Generalization: Bagging tends to improve the model's ability to generalize to unseen data by reducing overfitting. Each base learner is trained on a slightly different subset of the data, introducing diversity in the ensemble.

Parallelizable: Since each base learner is trained independently, bagging can be parallelized easily. This makes it suitable for scaling up on multi-core processors or distributed computing environments.

Compatibility with Any Base Learner: Bagging is a general ensemble technique that can work with any base learning algorithm. It does not require any modifications to the base learner, making it a versatile method.

Some popular variations and extensions of bagging include Random Forests, which use decision trees as base learners, and Out-of-Bag (OOB) estimation, which leverages the unused instances during bootstrap sampling for model evaluation. Overall, bagging is an effective ensemble method that can enhance the performance and robustness of machine learning models, making it a valuable tool in the field of predictive analytics.

1.4 MapReduce Scheme

MapReduce is a programming model and an associated implementation for processing and analyzing large-scale data sets in a parallel and distributed manner. It is commonly used in big data processing frameworks like Apache Hadoop. The MapReduce model divides the data processing task into two main stages: the map stage and the reduce stage.
Here's an overview of the MapReduce scheme:

Map Stage:
Input: The input data is divided into smaller chunks, and each chunk is assigned to a map function.

Map Function: The map function takes the input data chunk and performs a transformation on each element of the chunk, generating a set of intermediate key-value pairs.

Intermediate Key-Value Pairs: The map function outputs intermediate key-value pairs, where the key represents a specific category or grouping criterion, and the value is the processed data associated with that key.

Shuffle and Sort:
Partitioning: The intermediate key-value pairs are partitioned based on their keys so that pairs with the same key are sent to the same reducer.

Sorting: Within each partition, the intermediate key-value pairs are sorted based on their keys. This sorting is essential for the reduce stage.

Reduce Stage:
Reduce Function: Each reducer receives a subset of the intermediate key-value pairs, grouped by key. The reduce function takes these pairs as input and performs aggregation or other computations on the values associated with each key.

Output: The reduce function produces the final output, which is typically a reduced set of key-value pairs or a summary of the processed data.

The MapReduce scheme allows for parallel processing and distributed computation, which enables efficient processing of large datasets across multiple machines or nodes in a cluster. The map stage processes data in parallel, with each map function working on a separate data chunk. The intermediate key-value pairs are then shuffled and sorted to ensure that pairs with the same key are sent to the same reducer. Finally, the reduce stage aggregates the data for each key independently, producing the final result.

MapReduce is designed to handle fault tolerance and scalability, as it can handle failures of individual map or reduce tasks and automatically rerun them on other nodes. It also supports data locality, meaning that the processing tasks are performed on the same node where the data resides, reducing network overhead. Overall, the MapReduce scheme provides a powerful and scalable framework for processing large datasets by dividing the computation into smaller tasks and distributing them across a cluster of machines, enabling efficient big data processing and analysis.

2. Existing System

In 1950, Alan Turing introduced the concept of the Turing Test as a benchmark to determine the intelligence of machines. According to this test, a machine would be considered "intelligent" if it could successfully convince a human evaluator that it was also a human. This test laid the foundation for the field of Artificial Intelligence (AI) and became a significant milestone in its development. Shortly after, in the summer of 1956, a research program held at Dartmouth College became known as the birthplace of AI. This program brought together leading researchers in the field and marked the official beginning of AI as a distinct discipline. From this point forward, researchers started exploring and developing "intelligent" machine learning algorithms and computer programs. These early AI systems showcased a wide range of capabilities. For instance, some algorithms were designed to plan optimal travel routes for salespeople, while others were developed to play strategic board games like checkers and tic-tac-toe against human opponents. Over time, AI systems continued to advance and demonstrate remarkable achievements. They began to employ speech recognition techniques, enabling them to learn and pronounce words similar to how a baby would learn to speak. Furthermore, AI systems reached unprecedented milestones, such as defeating a world chess champion in a game that was once considered the pinnacle of human intellect. The evolution of machine learning can be visualized through an infographic that showcases its history. This timeline illustrates the progression of machine learning from its early stages, characterized by mathematical models, to the sophisticated and powerful technology it has become today.

3. Literature Review

Telecom churn was predicted using several methods. Most of these methods employed machine learning and data mining. Most related study employed one data mining approach to extract information, whereas others compared churn prediction methodologies. Gavril et al. [12] used 21 characteristics and a dependent churn parameter with two values to predict prepaid customer attrition. Features include customer voicemail and message counts. PCA reduced data dimensionality. Neural Networks, Support Vector Machines, and Bayes Networks predicted churn. Algorithm performance was measured using AUC. Bayes, neural, and support vector machine AUCs were 99.10%, 99.55%, and 99.70%, respectively. This research employed a short dataset without missing values. He et al. [13] used the Neural Network technique to forecast customer turnover in a big Chinese telecom firm with 5.23 million users. Overall forecast accuracy was 91.1%. Idris [14] used genetic programming with AdaBoost to simulate telecom churn. Two standard datasets tested the model. Orange Telecom's dataset was 63% accurate, whereas cell2cell's was 89%. Huang et al. [15] examined big data customer turnover. The researchers wanted to show that volume, diversity, and velocity of big data improve churn prediction. China's biggest telecoms firm needs a big data platform to create cracks in Operation Support and Business Support data. AUC assessed Random Forest method. Makhtar et al. [16] used crude set theory to forecast telecom churn. In this article, Rough Set classification beat Linear Regression, Decision Tree, and Voted Perception Neural Network. As a key challenge in churn prediction, imbalanced data sets with smaller churned customer classes than active customer classes were explored. Amin et al. [17] examined six telecom churn prediction oversampling methods. MTDF and genetic-algorithm-based rules-generation beat the other oversampling techniques. Burez and Van den Poel [8] examined Random Sampling, Advanced Under-Sampling, Gradient Boosting Model, and Weighted Random Forests for churn prediction models using unbalanced datasets. (AUC, Lift) metrics assessed the model under sampling fared best. No Syrian telecom firm has study on this issue. Most earlier research publications used telecom company or internet-published features instead of feature engineering or building features from raw data. This study considers feature engineering to develop features for machine learning algorithms. A big data platform prepared the data and compared four tree-based machine learning methods.

4. Proposed System

Consumer attribute analysis using machine learning algorithms is a valuable technique for gaining insights into consumer behavior and preferences. By leveraging machine learning, businesses can extract patterns, identify key attributes, and make data-driven decisions to enhance marketing strategies, product development, and customer satisfaction. Here are some common machine learning algorithms used for consumer attribute analysis:

Clustering Algorithms: Clustering algorithms like k-means, hierarchical clustering, or DBSCAN can group consumers based on similarities in their attributes. This helps identify distinct customer segments with similar preferences, behaviors, or demographics.

Decision Trees: Decision trees can be used to analyze consumer attributes and create rules that determine consumer behavior or preferences. This algorithm can help identify the most relevant attributes that influence consumer decisions.

Random Forests: Random forests combine multiple decision trees to improve accuracy and reduce overfitting. This algorithm can be used to predict consumer behavior, such as purchasing decisions or product preferences, based on their attributes.

Support Vector Machines (SVM): SVM is a supervised learning algorithm that can classify consumers based on their attributes. It can help businesses identify different consumer groups or predict specific behaviors based on historical data.

Association Rule Mining: Association rule mining, including algorithms like Apriori or FP-growth, can discover relationships and associations between consumer attributes. This technique is useful for market basket analysis, where businesses can identify which attributes or products are frequently purchased together.

Collaborative Filtering: Collaborative filtering algorithms, such as item-based or user-based collaborative filtering, can provide personalized recommendations to consumers based on their attributes and preferences. This technique is commonly used in recommendation systems.

Neural Networks: Neural networks, including deep learning models like convolutional neural networks (CNNs) or recurrent neural networks (RNNs), can analyze consumer attributes to make predictions or extract complex patterns. These models are particularly useful when dealing with large-scale consumer data or unstructured data like images or text. It's important to note that the choice of algorithm depends on the specific problem and data available. Exploratory data analysis, feature engineering, and model evaluation are crucial steps in consumer attribute analysis to ensure accurate and meaningful insights are obtained.

5. Implementation

5.1 Churn Prediction

Once the model is trained and validated, it can be deployed to predict churn for new customers or existing ones. By inputting customer attributes and behavioral data into the model, businesses can obtain churn probability scores for each customer. These scores indicate the likelihood of a customer churning in the near future.

Identification of Churn Risk Factors: Statistical modeling allows businesses to identify the key factors contributing to customer churn. By examining the model's coefficients or feature importances, businesses can understand which variables have the most significant impact on churn. This information helps prioritize efforts to address specific risk factors.

Development of Retention Strategies: Armed with insights from the statistical model, businesses can develop targeted retention strategies. For example, if the model identifies low customer satisfaction as a key churn risk factor, businesses can focus on improving customer service or launching loyalty programs to enhance satisfaction levels.

Monitoring and Evaluation: Implementing churn reduction strategies is an ongoing process. Businesses should continuously monitor customer churn rates, track the effectiveness of their retention efforts, and reevaluate the statistical model periodically. This allows for timely adjustments and improvements to the churn reduction strategies.

Iterative Improvement: The churn reduction process should be iterative, leveraging continuous learning from customer data and refining the statistical model over time. As more data becomes available and new insights are gained, the model can be updated to further improve churn prediction accuracy and identify additional churn risk factors.

5.2 CHURN ANALYSIS AND MAPREDUCE

Churn analysis and MapReduce are two concepts that can be combined to handle large-scale data processing and churn prediction tasks efficiently. Here's an overview of how MapReduce can be applied in churn analysis:

Churn Analysis: Churn analysis involves examining customer data to identify patterns and factors that contribute to customer attrition. This analysis typically requires processing and analyzing large volumes of data, such as customer demographics, transaction history, interactions, and other relevant attributes.

MapReduce: MapReduce is a programming model and associated implementation commonly used for distributed data processing. It enables the parallel processing of large datasets across a cluster of computers, making it suitable for handling big data tasks. MapReduce divides data processing into two stages: the map stage and the reduce stage.

Map Stage: In the map stage, data is divided into smaller chunks and processed independently by multiple nodes in a distributed computing environment. Each node applies a specified operation (map function) to the input data and generates intermediate key-value pairs.

Reduce Stage: In the reduce stage, the intermediate results produced by the map stage are combined and aggregated to produce the final output. Nodes perform another operation (reduce function) to process and consolidate the intermediate data, generating the desired result.

Applying MapReduce to Churn Analysis: In churn analysis, MapReduce can be applied to process and analyze large customer datasets efficiently. The steps involved in using MapReduce for churn analysis may include:

Data Partitioning: The customer data is divided into smaller chunks (splits) that can be processed in parallel by different nodes.

Map Function: Each node applies a map function to process its assigned data chunk independently. This function can involve filtering, transforming, or extracting relevant features from the customer data.

Intermediate Key-Value Pairs: The map function generates intermediate key-value pairs based on the processed data. The key might be a customer identifier, and the value could be a churn indicator or any relevant information.

Shuffling and Sorting: The intermediate key-value pairs are shuffled and sorted to group related data together, enabling efficient data consolidation in the reduce stage.

Reduce Function: The reduce function takes the shuffled data as input, performs further analysis or aggregation, and generates the final churn analysis results. This could include calculating churn rates, identifying churn risk factors, or generating churn prediction models.

Scalability and Performance Benefits: MapReduce offers inherent scalability and performance benefits when processing large datasets. By distributing the data processing across multiple nodes, it allows for parallelization, reducing the overall processing time. It also enables fault tolerance, as the system can continue processing even if some nodes fail.

By utilizing the MapReduce framework, businesses can efficiently perform churn analysis on vast customer datasets, extract valuable insights, and make informed decisions to mitigate customer churn. The parallel processing capabilities of MapReduce enable faster and more scalable churn analysis, supporting data-driven strategies for customer retention and business growth.

5.3 IDENTIFY CHURN RISK FACTORS

Leveraging statistical modeling and predictive analytics empowers businesses to identify churn risk factors effectively. By analyzing customer data and employing advanced modeling techniques, businesses can uncover patterns and correlations that contribute to customer churn. Here's how statistical modeling and predictive analytics help identify churn risk factors:

Data Analysis: Statistical modeling starts with comprehensive data analysis. By examining historical customer data, businesses can identify variables and attributes that are likely to influence churn. This may include customer demographics, purchase behavior, engagement metrics, customer service interactions, and more.

Feature Selection: Once the relevant data is identified, feature selection techniques are applied to choose the most informative variables. This process helps narrow down the attributes that have the most significant impact on churn prediction. Techniques like correlation analysis, mutual information, or feature importance from machine learning models can aid in feature selection.

Model Development: Statistical modeling techniques such as logistic regression, decision trees, random forests, support vector machines, or neural networks are employed to build predictive models. These models are trained using historical data, where churners and non-churners are labeled, and the models learn to predict churn based on customer attributes.

Model Evaluation: The trained models are evaluated using evaluation metrics such as accuracy, precision, recall, or area under the curve (AUC). The models' performance is assessed on a holdout dataset or through cross-validation to ensure their ability to generalize to unseen data.

Interpretation of Results: Statistical modeling provides insights into the relative importance and impact of different variables on churn. By examining the model coefficients, feature importances, or permutation importance, businesses can identify the key churn risk factors that significantly influence customer attrition.

Actionable Insights: Armed with the knowledge of churn risk factors, businesses can take proactive measures to mitigate churn. Strategies can be devised to address specific risk factors, such as improving customer service, enhancing product offerings, refining pricing strategies, or personalizing communication to retain at-risk customers.

Ongoing Monitoring and Refinement: Churn risk factors may evolve over time, and customer behavior patterns can change. Therefore, it is essential to continuously monitor and update the models as new data becomes available. This allows businesses to adapt their retention strategies and refine their predictive models for improved accuracy.

By leveraging statistical modeling and predictive analytics, businesses gain a deeper understanding of customer churn and can take targeted actions to reduce churn rates. This data-driven approach enables businesses to allocate resources effectively, enhance customer retention strategies, and ultimately improve customer satisfaction and long-term profitability.

5.4 Development

To develop the churn predictive system at SyriaTl, the installation of a big data platform is necessary. The chosen platform for this purpose is Horton works Data Platform (HDP), primarily due to its availability as a free and open-source framework. An additional advantage is that it operates under the Apache 2.0 License. The HDP platform encompasses a diverse range of open-source systems and tools that are relevant to big data. These tools and systems are interconnected and integrated with one another. The ecosystem of HDP can be visualized through Figure 1, which illustrates the categorization of tools into specific specializations such as Data Management, Data Access, Security, Operations, and Governance Integration.

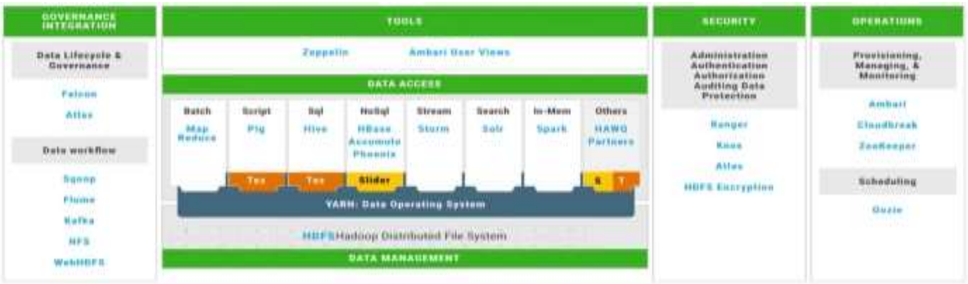

Figure 1 Hortonworks data platform HDP—big data framework

To streamline the installation process and ensure the inclusion of only the necessary tools and systems for all phases of the project, the HDP[1] framework was customized. This tailored package of installed systems and tools is referred to as the SYTL-BD framework (SyriaTel's big data framework). Within the SYTL-BD framework, we specifically installed the following components:

- Hadoop Distributed File System (HDFS)[2]: This was set up to serve as the data storage solution.
- Spark execution engine[3]: Utilized for data processing tasks.
- Yarn[4]: Employed for resource management within the framework.
- Zeppelin[5]: Chosen as the development user interface for enhanced productivity.
- Ambari[6]: Implemented for system monitoring purposes.
- Ranger[7]: Utilized to ensure the security of the system.
- Flume[8] System and Scoop tool[9]: Integrated to enable data acquisition from external sources into HDFS within the SYTL-BD framework.

In terms of hardware resources, we deployed a total of 12 nodes, each equipped with 32 Gigabytes of RAM, a storage capacity of 10 Terabytes, and a 16-core processor. To facilitate the churn predictive model development, a dataset spanning nine consecutive months was collected. This dataset serves as the basis for extracting the relevant features for the churn prediction model. Throughout the data lifecycle, the dataset underwent multiple stages, which are illustrated in Figure 2.

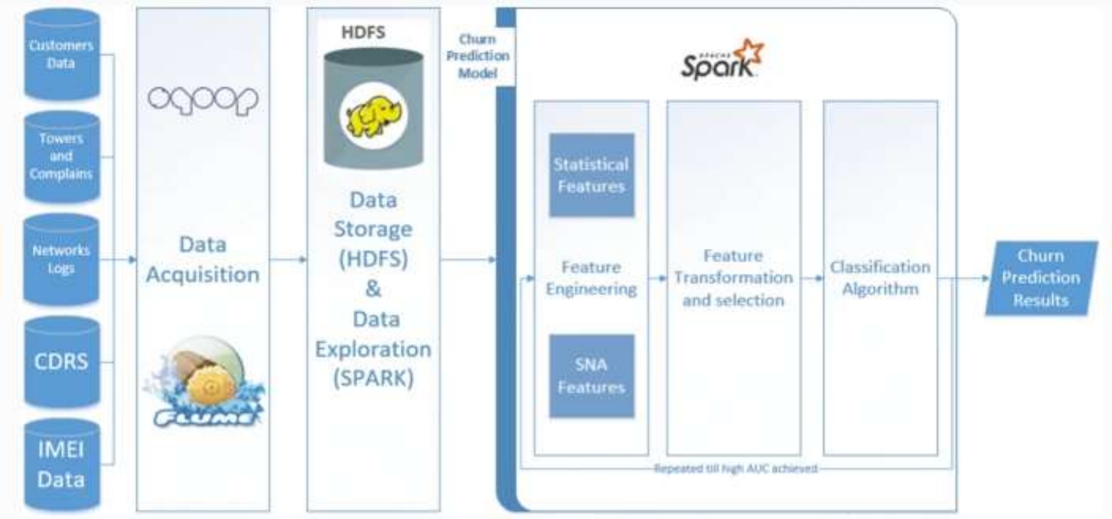

Figure 2 Proposed Churn Prediction System Architecture

The Spark engine played a crucial role in multiple phases of the churn predictive model, including data processing, feature engineering, model training, and model testing. Its utilization was primarily driven by its ability to perform processing tasks in memory (RAM). This in-memory processing capability provided significant advantages for the project. Furthermore, the Spark engine offers numerous additional benefits. One notable advantage is its extensive collection of libraries, which encompass a wide range of functionalities required for implementing various stages of the machine learning lifecycle. These libraries provide convenient and efficient solutions for tasks such as data manipulation, statistical analysis, feature extraction, model training, and evaluation. The availability of such a comprehensive set of libraries within the Spark engine greatly simplifies and accelerates the development process of the churn predictive model.

The initial stage of the project involved transferring data from external sources into HDFS within the SYTL-BD framework. The data was categorized into three primary types: structured, semi-structured, and unstructured. To facilitate this process, Apache Flume, a distributed system, was employed. Flume was responsible for collecting and transporting unstructured data files (such as CSV and text) as well as semi-structured data files (including JSON and XML) to HDFS. Figure 3 illustrates the architectural design of Flume within the SYTL-BD framework. Flume consists of three main components: the data source, the channel through which the data flows, and the sink where the data is ultimately delivered. Together, these components form the necessary infrastructure for seamless and efficient data collection and movement within the SYTL-BD framework.

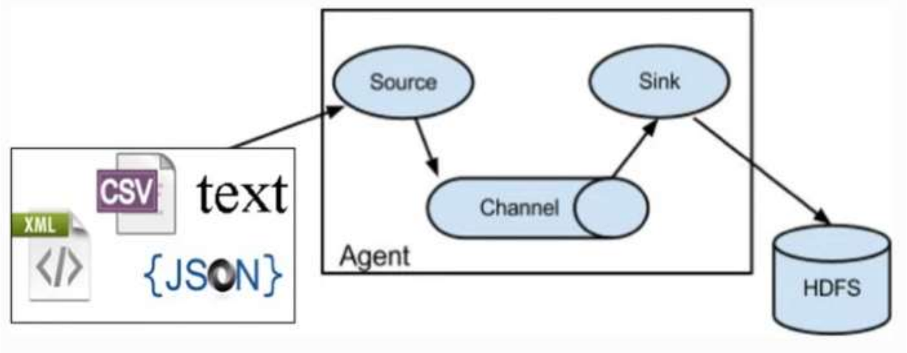

Figure 3 Apache Flume Configured System Architecture

Within the SYTL-BD framework, the Flume agents responsible for file transportation are configured to operate within a defined Spooling Directory Source. These agents utilize a single channel, which has been specifically configured as a Memory Channel. This channel has demonstrated superior performance compared to other available channels in Flume. As data flows through this channel, it is ultimately written to the designated sink, which in this case is HDFS. Importantly, the data retains its original format throughout the transformation process and subsequent storage in HDFS. To facilitate the transfer of structured data from relational databases, Apache Sqoop, a distributed tool, is employed. This tool enables the bulk transfer of data between HDFS and relational databases. Using Map jobs, Sqoop efficiently moves all the data from the databases to HDFS. The architecture of the Sqoop import process is depicted in Figure 4, where four mappers are defined by default. Each Map job is responsible for selecting a portion of the data and transferring it to HDFS. Upon transportation by Sqoop to HDFS, the data is saved in the CSV file format.

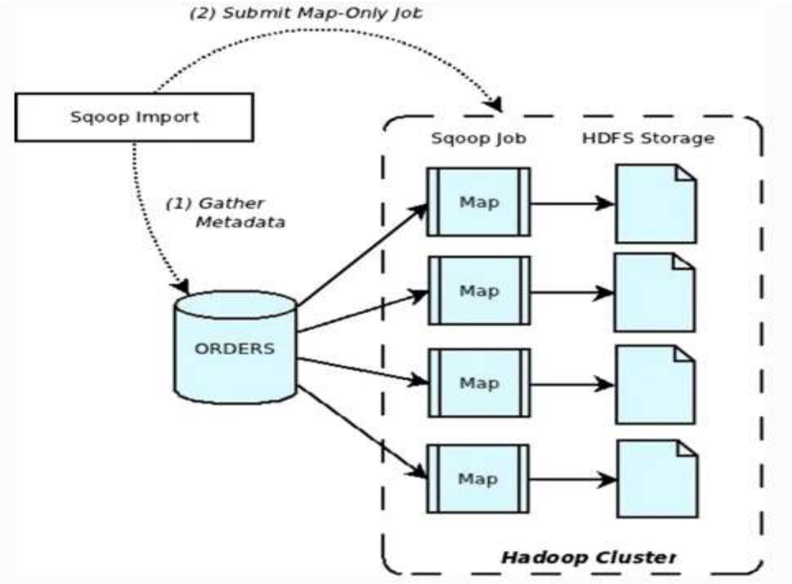

Figure 4 Apache SQOOP data import architecture.

Once all the data from various sources was successfully transported into HDFS, it became crucial to determine the optimal file type that would maximize performance in terms of both space utilization and execution time. To conduct this experiment, the Spark engine was utilized, specifically leveraging the Data Frame library[10].

The experiment involved transforming a massive 1 terabyte CSV dataset into two different file types: Apache Parquet[11] and Apache Avro[12]. These file types were chosen based on their potential to deliver superior performance characteristics. Additionally, three compression scenarios were considered during the experiment. Compression plays a significant role in reducing the storage space required for the dataset and improving overall processing efficiency. By evaluating the performance of different file types and compression options, the aim was to identify the combination that offered the best trade-off between space utilization and execution time.

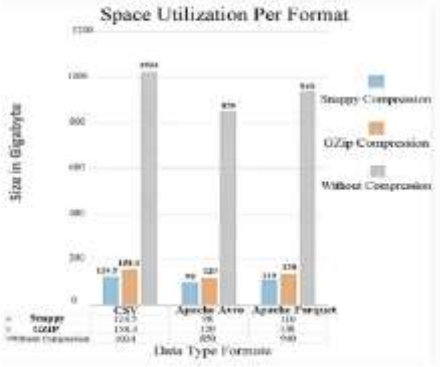

Figure 5 Space Utilization per Format

Upon successfully transferring all the data from its sources into HDFS, the next crucial step was to select the optimal file type that would provide the best performance in terms of space utilization and execution time. To accomplish this, the experiment was conducted using the Spark engine, leveraging the capabilities of the Data Frame library. The experiment involved transforming a massive 1 terabyte dataset in CSV format into two different file types: Apache Parquet [11] and Apache Avro[12]. These file types were chosen based on their potential to deliver superior performance characteristics.

Furthermore, the experiment considered three different compression scenarios. The choice of compression technique plays a significant role in reducing storage space requirements and enhancing overall processing efficiency. By assessing the performance of various file types and compression options, the aim was to identify the combination that achieved the optimal balance between space utilization and execution time.

6. Conclusion

The study of the telecom industry is quite important for assisting businesses in making more money. It is well known that one of the most important aspects of a telecom company's financial performance is the ability to forecast churn. As a result, the goal of this study was to create a churn prediction system that was especially designed for the telecom business SyriaTel. High AUC (Area Under the Curve) values were essential for the prediction models to produce accurate churn forecasts. A training set containing 70% of the data and a testing set containing the remaining 30% made up the dataset. A 10-fold cross-validation method was used to achieve accurate model performance assessment and hyper parameter adjustment. The features were prepared and transformed using a variety of approaches, such as feature engineering and efficient feature selection techniques, to make them appropriate for machine learning algorithms. The data's imbalance, with just around 5% of the records indicating consumers who had left, presented another difficulty. Under sampling and using tree-based techniques, which are less impacted by class imbalance, were used to overcome this problem. Due to their variety and suitability for churn prediction, four tree-based algorithms were selected: Decision Tree, Random Forest, Gradient Boosting Machine, and XGBOOST (Extreme Gradient Boosting). The model's success was greatly influenced by the planning, choice, and inclusion of mobile social network elements. Notably, the SyriaTel dataset's AUC score for the XGBOOST method was 93.301%, making it the top-performing model across all tests. In terms of AUC values, the GBM algorithm came in second, followed by the Random Forest and Decision Tree algorithms. Without taking any actively marketing measures, a fresh dataset from various time periods was fitted to the models for further assessment. With an AUC of 89%, the XGBOOST algorithm once again showed its excellent performance. The non-stationary character of the data may be to blame for the minor decline in outcomes, underscoring the need for regular model retraining. The results of churn prediction in the telecom sector were improved by the addition of Social Network Analysis components.

References

1. Gerpott TJ, Rams W, Schindler A. Customer retention, loyalty, and satisfaction in the German mobile cellular telecommunications market. Telecommun Policy. 2001;25:249–69.
2. Wei CP, Chiu IT. Turning telecommunications call details to churn prediction: a data mining approach. Expert Syst Appl. 2002;23(2):103–12.
3. [3] Qureshii SA, Rehman AS, Qamar AM, Kamal A, Rehman A. Telecommunication subscribers’ churn prediction model using machine learning. In: Eighth international conference on digital information management. 2013. p. 131–6.
4. [4] Ascarza E, Iyengar R, Schleicher M. The perils of proactive churn prevention using plan recommendations: evidence from a field experiment. J Market Res. 2016;53(1):46–60.
5. [5] Bott. Predicting customer churn in telecom industry using multilayer preceptron neural networks: modeling and analysis. Igarss. 2014;11(1):1–5.
6. [6] Umayaparvathi V, Iyakutti K. A survey on customer churn prediction in telecom industry: datasets, methods and metric. Int Res J Eng Technol. 2016;3(4):1065–70.
7. [7] Yu W, Jutla DN, Sivakumar SC. A churn-strategy alignment model for managers in mobile telecom. In: Communication networks and services research conference, vol. 3. 2005. p. 48–53.
8. [8] Burez D, den Poel V. Handling class imbalance in customer churn prediction. Expert Syst Appl. 2009;36(3):4626–36.
9. [9] Zhan J, Guidibande V, Parsa SPK. Identification of top-k influential communities in big networks. J Big Data. 2016;3(1):16. https://doi.org/10.1186/s40537-016-0050-7.
10. [10] Barthelemy M. Betweenness centrality in large complex networks. Eur Phys J B. 2004;38(2):163–8. https://doi.org/10.1140/epjb/e2004-00111-4.
11. [11] Elisabetta E, Meyerhenke H, Staudt CL. Approximating betweenness centrality in large evolving networks. CoRR. 2014. arxiv:1409.6241.
12. [12] Brandusoiu I, Toderean G, Ha B. Methods for churn prediction in the prepaid mobile telecommunications industry. In: International conference on communications. 2016. p. 97–100.
13. [13] He Y, He Z, Zhang D. A study on prediction of customer churn in fixed communication network based on data mining. In: Sixth international conference on fuzzy systems and knowledge discovery, vol. 1. 2009. p. 92–4.

14. [14] Idris A, Khan A, Lee YS. Genetic programming and adaboosting based churn prediction for telecom. In: IEEE international conference on systems, man, and cybernetics (SMC). 2012. p. 1328–32.
15. [15] Huang F, Zhu M, Yuan K, Deng EO. Telco churn prediction with big data. In: ACM SIGMOD international conference on management of data. 2015. p .607–18.
16. [16] Makhtar M, Nafis S, Mohamed M, Awang M, Rahman M, Deris M. Churn classification model for local telecommunication company based on rough set theory. J Fundam Appl Sci. 2017;9(6):854–68.
17. [17] Amin A, Anwar S, Adnan A, Nawaz M, Howard N, Qadir J, Hawalah A, Hussain A. Comparing oversampling techniques to handle the class imbalance problem: a customer churn prediction case study. IEEE Access. 2016;4:7940–57.
18. [18] Chawla N. Data mining for imbalanced datasets: an overview. In: Data mining and knowledge discovery handbook. Berlin: Springer; 2005. p. 853–67.
19. [19 Page L, Brin S, Motwani R, Winograd T. The pagerank citation ranking: bringing order to the web. Stanford Digital Library Technologies Project. 1998. p. 17.
20. [20] Kiss C, Bichler M. Identification of influencers—measuring influence in customer networks. Decis Support Syst. 2008;46(1):233–53.
21. [21] Kiss C, Bichler M. Identification of influencers—measuring influence in customer networks. Decis Support Syst. 2008;46(1):233–53. https://doi.org/10.1016/j.dss.2008.06.007.
22. [22] Brin S, Page L. The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst. 1998;30(1–7):107–17. https://doi.org/10.1016/S0169-7552(98)00110-X.
23. [23] Zhao Y, Wang G, Yu PS, Liu S, Zhang S. Inferring social roles and statuses in social networks. In: KDD 2013—19th ACM SIGKDD international conference on knowledge discovery and data mining. Association for Computing Machinery; 2013. p. 695–703.
24. [24] Leskovec J, Backstrom L, Kumar R, Tomkins A. Microscopic evolution of social networks. In: International conference on knowledge discovery and data mining. KDD; 2008. p. 695–703.
25. [25] Li Y, Luo P, Wu C. A new network node similarity measure method and its applications. 2014. arxiv:14034303.
26. [26] Xie J, Rojkova V, Pal S, Coggeshall S. A combination of boosting and bagging for kdd cup 2009—fast scoring on a large database. J Mach Learn Res Proc Track. 2009;7:35–43.

27. [27] Chen T, Guestrin C. Xgboost. A scalable tree boosting system. CoRR. 2016. arXiv:1603.02754