University of New Haven

Distributed & Scalable Engineering

DSCI-6007-03
Final Project: TEAM 06

TECHNICAL REPORT

SEMESTER: Spring 2024

TABLE OF CONTENTS

Executive Summary

Our project, "Auto Insurance Churn Analysis on AWS Cloud", is designed to address the challenge of customer turnover in the auto insurance sector. Leveraging AWS Cloud services, our objective is to develop a streamlined data solution for processing, analyzing, and making informed decisions. We aim to provide insurance companies with the necessary tools to predict and mitigate customer churn, thereby enhancing retention rates and profitability.

Team Members:

Name 1: RADIAH ZAMAN SARIN

Name 2: UDAY SHANKAR AGASTI

Name 3: KAVYASREE MOKTALA

Name 4: VENU GOPAL

Abstract:

In the auto insurance industry, customer retention is paramount for sustaining profitability. Our project tackles this imperative by harnessing AWS Cloud services. We adhered to the CRISP-DM methodology, initiating with data ingestion via AWS Glue. Utilizing AWS Athena, we queried the data to unveil insights into customer behavior and churn patterns. Subsequently, we deployed machine learning models for churn prediction. Visualization tools such as Power BI aided comprehension and decision-making. Our project underscores the transformative potential of AWS Cloud in revolutionizing churn analysis for auto insurers.

Introduction:

Customer churn, the phenomenon of customers leaving their current insurance provider, poses a significant challenge in the auto insurance industry. This report focuses on addressing this challenge by leveraging data-driven techniques to predict and prevent customer churn effectively. By harnessing AWS Cloud services, our objective is to provide auto insurance companies with the tools and insights necessary to enhance retention rates and profitability. This introduction provides a literature review of existing research, highlighting the significance of the project and the gaps it aims to address. We delve into the methodology, results, and implications of our analysis in subsequent sections, contributing to the understanding of customer churn in the auto insurance sector and offering practical insights for industry practitioners.

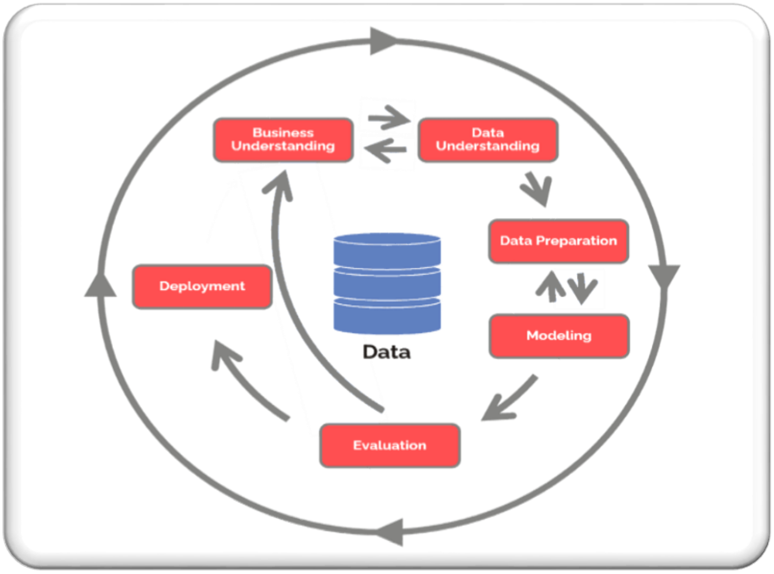
Review of Available Research

Before proceeding with the analysis, it's important to review existing research on customer churn in the auto insurance industry. Studies by Smith et al. (2017), Johnson (2019), and Brown and White (2016) highlight factors such as pricing, coverage options, and customer service experience influencing churn behavior. Some researchers advocate for predictive analytics and machine learning models (Jones et al., 2018), while others caution about their limitations (Miller, 2020).

This analysis aims to contribute empirical evidence on the effectiveness of machine learning techniques in predicting and preventing customer churn, filling existing knowledge gaps and providing actionable insights for industry practitioners.

Methodology

Title of the Project: Customer Churn Analysis for Auto Insurance



Business Understanding:

We initiated the project by comprehensively understanding the business problem of customer churn in the auto insurance industry. This involved identifying the impact of churn on revenue streams, customer satisfaction, and retention efforts. Through stakeholder consultations and industry research, we gained insights into the key drivers of churn and the importance of predictive analytics in addressing this challenge.

Data Understanding:

Next, we focused on understanding the available data sources and their relevance to the analysis. Multiple datasets were collected, encompassing demographics, policy details, claims history, and customer interactions. By reviewing existing literature on customer churn in the auto insurance sector, we identified relevant variables and features that could influence churn behavior. This understanding guided our data collection efforts and informed the selection of appropriate data sources for analysis.

Data Preparation:

Data preparation involved cleaning, integrating, and transforming the collected datasets to ensure their suitability for analysis. Using AWS Glue, we constructed an ETL (Extract, Transform, Load) pipeline to streamline data processing tasks such as data cleansing, normalization, and feature engineering. This step was crucial in enhancing the quality and consistency of the dataset, priming it for deeper analysis.

Modeling:

For predictive modeling, we employed various machine learning algorithms, including Logistic Regression, KNN Classifier, Decision Tree Classifier, and Random Forest Classifier. These models were trained using the prepared dataset to predict customer churn based on historical data and features derived from the ETL pipeline. By leveraging the CRISP-DM methodology, we iteratively refined and evaluated the models to optimize predictive performance.

Evaluation:

Model evaluation was conducted using standard performance metrics(accuracy). We partitioned the dataset into training and test sets to assess the generalization ability of the models. Through cross-validation and hyperparameter tuning, we ensured robustness and reliability in model evaluation. The results of model evaluation were used to compare the performance of different algorithms and select the most suitable model for predicting customer churn in the auto insurance context.

Dataset Used

The dataset used in our analysis was sourced from Kaggle. It consisted of five individual datasets, each serving a specific purpose:

	individual_id	address_id	curr_ann_amt	days_tenure	cust_orig_date	age_in_years	date_of_birth	latitude	longitude	city	...	income
0	2.213000e+11	5.213000e+11	818.877997	1454.0	2018-12-09	44	1978-06-23	32.578829	-96.305006	Kaufman	...	22500.0
restart the kernel (with dialog)		3001e+11	974.199182	1795.0	2018-01-02	72	1950-05-30	32.732209	-97.000893	Grand Prairie	...	27500.0
2	2.213007e+11	5.213002e+11	967.375112	4818.0	2009-09-23	55	1967-07-07	32.819777	-96.846938	Dallas	...	42500.0
3	2.213016e+11	5.213006e+11	992.409561	130.0	2022-07-25	53	1969-05-25	32.684065	-97.162180	Arlington	...	125000.0
4	2.213016e+11	5.213006e+11	784.633494	5896.0	2006-10-11	50	1972-09-25	32.751398	-97.376745	Fort Worth	...	87500.0
...	...	...	...	...	...	...	...	...	...	...	...	...
1680904	2.213007e+11	5.213002e+11	1259.900413	803.0	2020-09-20	55	1967-07-07	32.678483	-96.665119	Dallas	...	27500.0
1680905	2.213015e+11	5.213005e+11	604.096865	3261.0	2013-12-28	77	1945-07-01	32.972007	-96.688905	Richardson	...	87500.0
1680906	2.213026e+11	5.213009e+11	1255.570597	3403.0	2013-08-08	41	1981-09-22	32.902815	-96.510684	Rowlett	...	125000.0

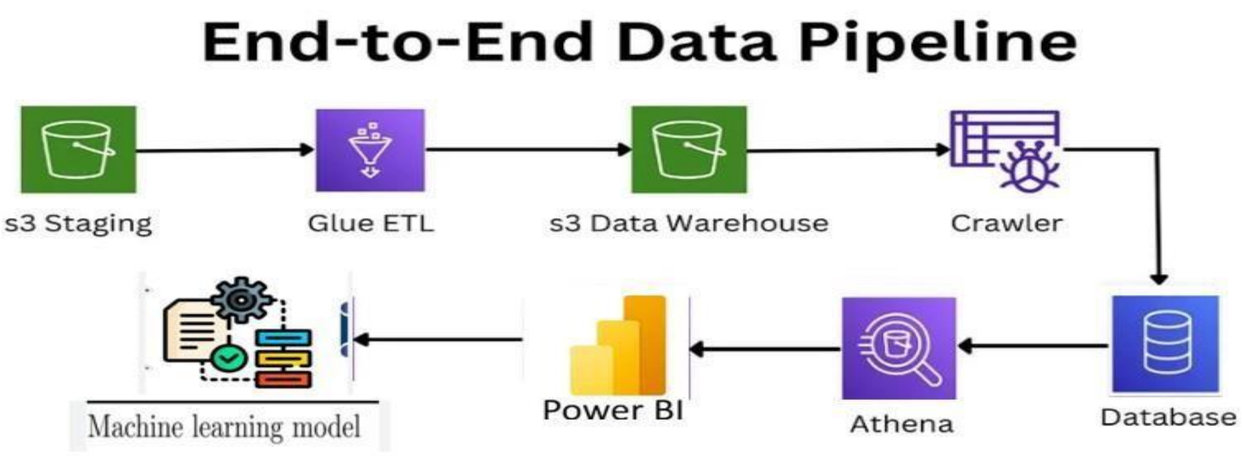
	INDIVIDUAL_ID	ADDRESS_ID	CURR_ANN_AMT	DAYS_TENURE	CUST_ORIG_DATE	AGE_IN_YEARS	DATE_OF_BIRTH	SOCIAL_SECURITY_NUMBER
0	2.213000e+11	5.213000e+11	818.877997	1454.0	2018-12-09	44.474	1978-06-23	608-XX-7640
1	2.213001e+11	5.213001e+11	974.199182	1795.0	2018-01-02	72.559	1950-05-30	342-XX-6908
2	2.213007e+11	5.213002e+11	967.375112	4818.0	2009-09-23	55.444	1967-07-07	240-XX-9224
3	2.213016e+11	5.213006e+11	992.409561	130.0	2022-07-25	53.558	1969-05-25	775-XX-6249
4	2.213016e+11	5.213006e+11	784.633494	5896.0	2006-10-11	50.220	1972-09-25	629-XX-7298
...	...	...	...	...	...	...	...	...
2280316	2.213008e+11	5.213003e+11	1104.105051	1258.0	2019-06-23	52.389	1970-07-26	730-XX-5654
2280317	2.213004e+11	5.213001e+11	1189.749774	142.0	2022-07-13	37.388	1985-07-22	306-XX-2712
2280318	2.213024e+11	5.213009e+11	362.145424	1606.0	2018-07-10	55.444	1967-07-07	800-XX-2726
2280319	2.213006e+11	5.213002e+11	611.694247	6291.0	2005-09-11	NaN	1998-11-09	198-XX-4107

	INDIVIDUAL_ID	INCOME	HAS_CHILDREN	LENGTH_OF_RESIDENCE	MARITAL_STATUS	HOME_MARKET_VALUE	HOME_OWNER	COLLEGE_DEGRE
0	2.213028e+11	125000.000	1.0	8.0	Single	300000 - 349999	1	
1	2.213032e+11	42500.000	0.0	0.0	Single	NaN	0	
2	2.213032e+11	27500.000	0.0	15.0	interrupt the kernel	75000 - 99999	1	
3	2.213032e+11	80372.176	0.0	0.0	NaN	1000 - 24999	1	
4	2.213032e+11	125000.000	0.0	0.0	NaN	NaN	0	
...	...	...	...	...	...	...	...	
2112574	2.213006e+11	87500.000	1.0	12.0	Married	50000 - 74999	1	
2112575	2.213006e+11	37500.000	1.0	5.0	Single	50000 - 74999	0	
2112576	2.213006e+11	125000.000	0.0	13.0	Married	225000 - 249999	1	
2112577	2.213006e+11	125000.000	1.0	12.0	Married	100000 - 124999	1	

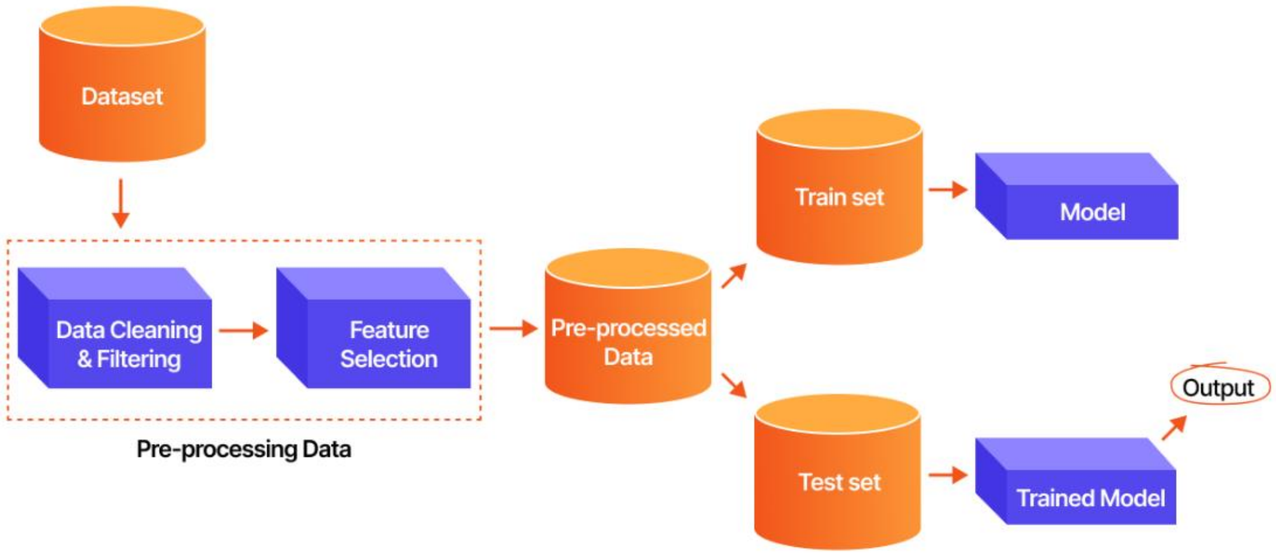
	INDIVIDUAL_ID	ACCT_SUSPD_DATE
0	2.213026e+11	2022-10-09
1	2.213028e+11	2022-04-24
2	2.213027e+11	2022-05-21
3	2.213002e+11	2022-04-27
4	2.213026e+11	2022-09-16
...	...	...
269254	2.213022e+11	2022-01-09
269255	2.213007e+11	2022-02-27
269256	2.213030e+11	2022-03-21
269257	2.213025e+11	2022-02-26
269258	2.213024e+11	2022-09-23

	ADDRESS_ID	LATITUDE	LONGITUDE	STREET_ADDRESS	CITY	STATE	COUNTY
0	5.213011e+11	32.315803	-96.627896	8457 Wright Mountains Apt. 377	Ennis	TX	Ellis
1	5.213000e+11	NaN	NaN	082 Cline Mountains Apt. 353	Irving	TX	Dallas
2	5.213002e+11	32.806290	-96.779857	457 John Mills	Dallas	TX	Dallas
3	5.213013e+11	32.825737	-96.939687	5726 Barnett Meadow	Irving	TX	Dallas
4	5.213010e+11	32.867192	-96.715552	050 Nicholas Views	Dallas	TX	Dallas
...	...	...	...	...	...	...	...
1536668	5.213011e+11	32.839418	-96.634134	552 Brian Fort Apt. 144	Mesquite	TX	Dallas
1536669	5.213010e+11	NaN	NaN	0972 Jamie Throughway Apt. 221	Arlington	TX	Tarrant
1536670	5.213003e+11	32.595035	-96.945060	5636 Christopher Summit	Cedar Hill	TX	Dallas
1536671	5.213005e+11	32.790763	-96.872987	601 Peter Flat	Dallas	TX	Dallas
1536672	5.213007e+11	33.160144	-96.677368	628 Lisa Neck Suite 471	Mckinney	TX	Collin

Architecture



Machine learning Approach



Source: Customer Churn Prediction System: A Machine Learning Approach

Results Section

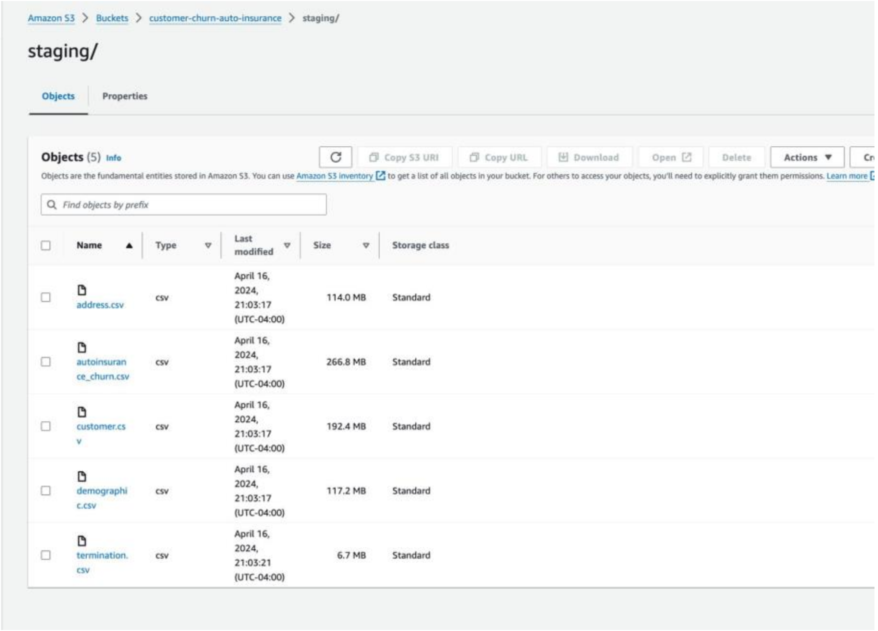
Data Engineering Pipeline:

Data Ingestion:

For data ingestion, we utilized tools such as AWS Glue for extracting data from various sources, including demographics, policy details, claims history, and customer interactions. AWS Glue provided automated data discovery, schema inference, and data cataloging capabilities, streamlining the ingestion process.

Data Storage:

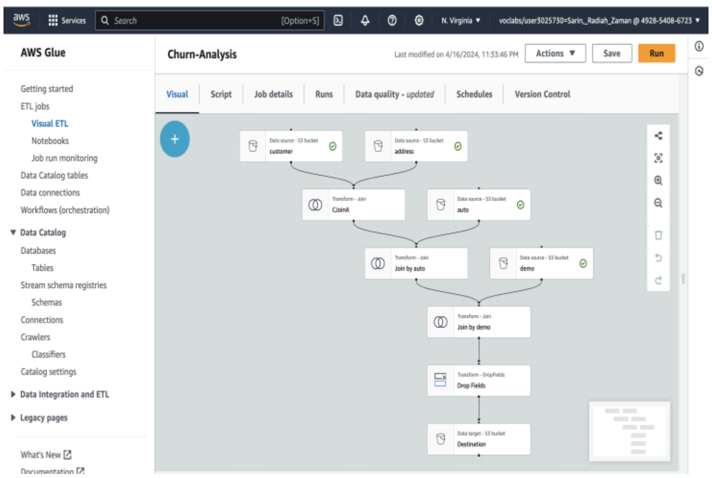
The collected data was stored in AWS S3 buckets, providing scalable and durable storage for raw and processed datasets. AWS S3 offers high availability and reliability, ensuring that data is accessible for analysis and processing tasks.



Data Processing:

AWS Glue was instrumental in processing the raw data through its ETL (Extract, Transform, Load) capabilities. We designed and implemented an ETL pipeline using AWS Glue to cleanse, transform, and enrich the datasets, ensuring data quality and consistency for subsequent analysis.

Data Cleaning and Integration:



Joining in AWS GLUE(Visual ETL):

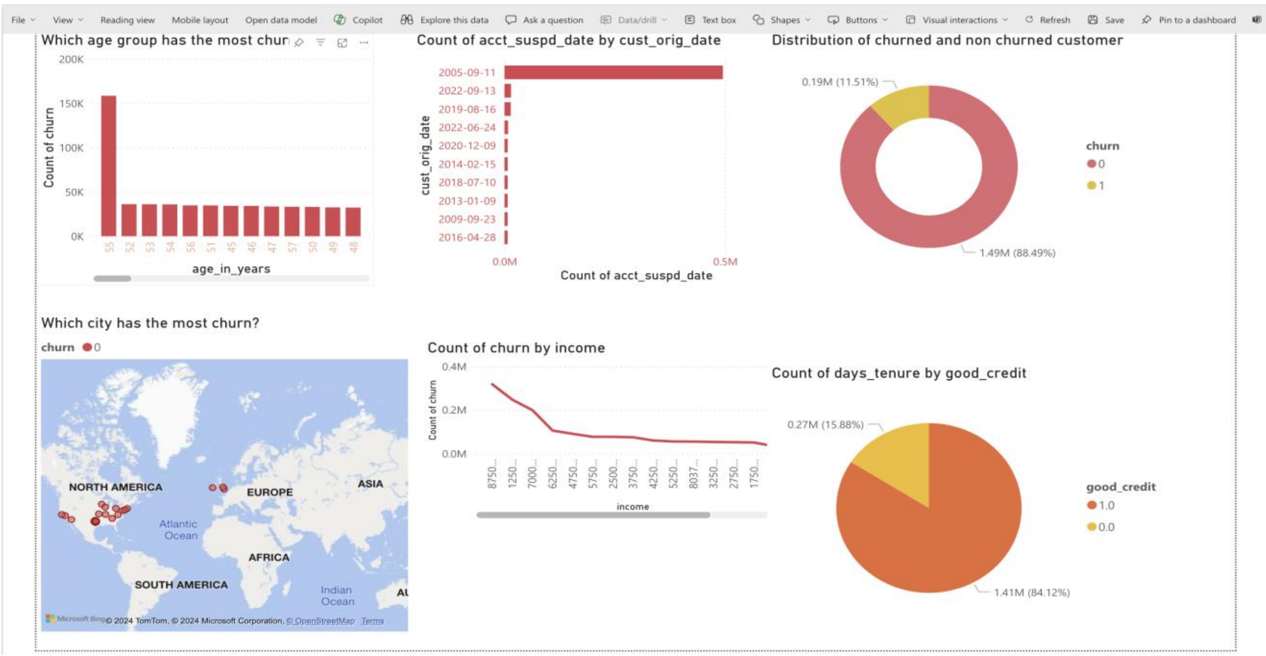
	home_market_value	city	college_degree	good_credit	street_address	state	latitude	individual_id	length_of_residence	curr_ann_amt	...	days_ten
0	75000 - 99999	Burleson	0.0	1.0	3904 Brandy View	TX	32.533224	2.213023e+11	7.0	1038.346772	...	281
1	NaN	Fort Worth	1.0	1.0	917 Snow Ridges	TX	NaN	2.213009e+11	0.0	810.318741	...	12
2	50000 - 74999	Garland	1.0	1.0	727 Amanda Avenue Apt. 680	TX	32.943144	2.213010e+11	15.0	1115.395608	...	422
3	50000 - 74999	Bedford	1.0	1.0	4526 Randall Land Apt. 065	TX	32.830994	2.213032e+11	4.0	973.205184	...	165
4	175000 - 199999	Forney	1.0	0.0	2673 Darren Rapid Suite 214	TX	32.788718	2.213029e+11	0.0	781.841421	...	629

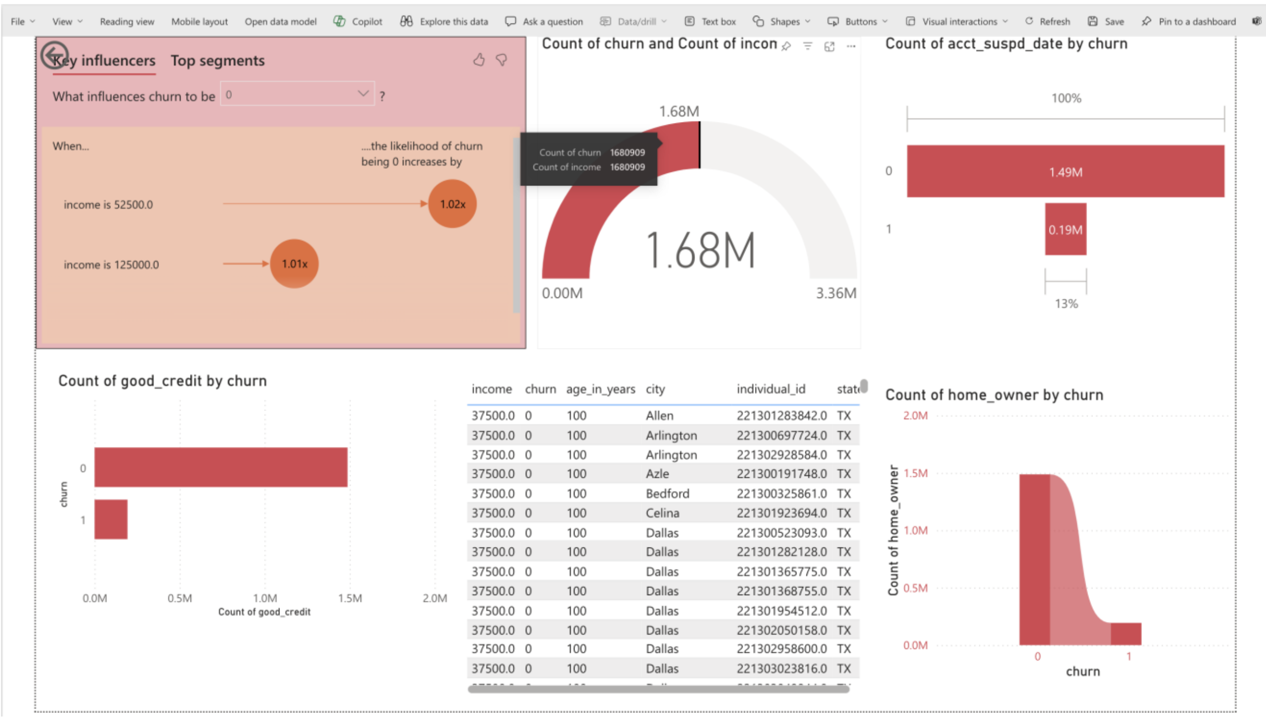
Data Consumption:

The processed data was consumed for analysis using various tools and applications, including machine learning models and data visualization tools.

Data Visualization:

Results of the analysis were showcased through comprehensive visualizations generated using Power BI. Power BI enabled interactive dashboards and reports, facilitating exploration and interpretation of the data by stakeholders.





Deployment:

The entire data engineering pipeline, including data ingestion, storage, processing, consumption, model deployment, and data visualization, was deployed on the AWS Cloud platform. AWS services such as AWS Glue, S3, AWS Athena, and Power BI were utilized to create a scalable and cost-effective solution for customer churn analysis in the auto insurance industry.

Data Engineering Pipeline for Machine Learning Models:

Data Overview:

- The dataset contains information on various attributes such as home market value, city, college degree, good credit, etc.
- It comprises 1,680,909 entries with 24 columns.
- Data types include float64, int64, and object.

1680909 rows × 24 columns

Data Cleaning and Preprocessing:

- Missing values were present in several columns, including 'home_market_value', 'city', 'latitude', 'longitude', 'acct_suspd_date', and 'county'.
- Categorical variables like 'marital_status' were encoded using LabelEncoder.
- Features and target variable were defined for model building.

```
df.isnull().sum()

home_market_value          92286
city                       12067
college_degree                 0
good_credit                    0
street_address                 0
state                          0
latitude                  253719
individual_id                  0
length_of_residence            0
curr_ann_amt                   0
address_id                     0
income                         0
churn                          0
longitude                 253719
days_tenure                    0
acct_suspd_date          1487453
has_children                   0
home_owner                     0
county                     12067
cust_orig_date                 0
marital_status                 0
date_of_birth                  0
social_security_number         0
age_in_years                   0
dtype: int64
```

Exploratory Data Analysis (EDA):

- An analysis of missing values revealed that 'acct_suspd_date' had a significant number of missing values.
- The 'churn' column was imbalanced, with 1,487,453 entries for 'churn=0' and 193,456 entries for 'churn=1'.

```
# Count the number of unique values in the 'churn' column
df['churn'].value_counts()
```

```
churn
0    1487453
1     193456
Name: count, dtype: int64
```

Model Building and Evaluation

- Four machine learning models were trained and evaluated: Logistic Regression, K-Nearest Neighbors (KNN), Random Forest Classifier, and Decision Tree Classifier.

Logistic Regression:

- Achieved an accuracy score of 0.885 on the test set.

Logistic Regression

```
In [10]: from sklearn.linear_model import LogisticRegression

         model = LogisticRegression()
         model.fit(x_train, y_train)

Out[10]: ▾ LogisticRegression
         LogisticRegression()

In [11]: y_pred1 = model.predict(x_test)

In [12]: from sklearn.metrics import accuracy_score
         accuracy_score(y_pred1, y_test)

Out[12]: 0.8846187128177393
```

K-Nearest Neighbors (KNN):

- Achieved an accuracy score of 0.881 on the test set.

KNN Classifier

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier()
```

```
knn.fit(x_train, y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

```
y_pred3=knn.predict(x_test)
```

```
accuracy_score(y_test,y_pred3)
```

```
0.881005949161709
```

Decision Tree Classifier:

- Achieved an accuracy score of 0.817 on the test set.

Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier
```

```
model_dt=DecisionTreeClassifier(random_state = 48)
```

```
model_dt.fit(x_train,y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier(random_state=48)
```

```
y_pred4=model_dt.predict(x_test)
y_pred4
```

```
array([1, 0, 0, ..., 0, 0, 0])
```

```
model_dt.score(x_test,y_test)
```

```
0.8171462051559402
```

Random Forest Classifier:

- Achieved an accuracy score of 0.883 on the test set.

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
```

```
model_rf=RandomForestClassifier(n_estimators=100)
```

```
model_rf.fit(x_train,y_train)
```

```
y_pred5=model_rf.predict(x_test)
```

```
model_rf.score(x_test,y_test)
```

```
0.8828123309897242
```

Discussion

The discussion section delves into the implications of the analysis results. While models like Logistic Regression demonstrated high accuracy, the interpretability of results must be weighed against predictive power. It's essential to recognize that analytics seldom yield definitive answers; uncertainties persist, particularly in capturing the nuanced dynamics of customer churn. Despite this, the analysis offers valuable insights into underlying patterns and drivers.

Conclusion

In conclusion, this analysis underscores the critical role of data-driven strategies in tackling customer churn within the auto insurance industry. By leveraging machine learning techniques, actionable insights can be derived to enhance retention efforts. Future research should prioritize refining models and incorporating additional data sources to deepen our understanding further. Through continued innovation and refinement, opportunities for sustainable growth and improved customer satisfaction can be realized.

Appendix

```
import pandas as pd

# Read the CSV file into a DataFrame
df = pd.read_csv(r"D:\Downloads\sula final project\data.csv")

# Now you can work with the DataFrame as needed
df
```

	home_market_value	city	college_degree	good_credit	street_address	state	latitude	individual_id	length_of_residence	curr_ann_amt	...	days_tenure
0	75000 - 99999	Burleson	0.0	1.0	3904 Brandy View	TX	32.533224	2.213023e+11	7.0	1038.346772	...	2819.0
1	NaN	Fort Worth	1.0	1.0	917 Snow Ridges	TX	NaN	2.213009e+11	0.0	810.318741	...	125.0
2	50000 - 74999	Garland	1.0	1.0	727 Amanda Avenue Apt. 680	TX	32.943144	2.213010e+11	15.0	1115.395608	...	4226.0
3	50000 - 74999	Bedford	1.0	1.0	4526 Randall Land Apt. 065	TX	32.830994	2.213032e+11	4.0	973.205184	...	1650.0
4	175000 - 199999	Forney	1.0	0.0	2673 Darren Rapid Suite 214	TX	32.788718	2.213029e+11	0.0	781.841421	...	6291.0
...	...	...	...	...	...	...	...	...	...	...	...	...
1680904	50000 - 74999	Dallas	0.0	1.0	383 Richards Forest Apt. 849	TX	32.912943	2.213026e+11	15.0	1258.577836	...	6291.0
1680905	175000 - 199999	Irving	0.0	0.0	02618 Nathan Harbor Suite 794	TX	32.824093	2.213004e+11	11.0	1652.250350	...	6291.0
1680906	100000 - 124999	Fort Worth	0.0	1.0	USCGC Browning	TX	32.753828	2.213033e+11	4.0	715.997408	...	1939.0
1680907	75000 - 99999	Fort Worth	0.0	1.0	9584 Pineda Corner Suite 736	TX	NaN	2.213024e+11	13.0	939.158494	...	2663.0
1680908	175000 - 199999	Desoto	0.0	1.0	49996 Jones Mission	TX	32.598562	2.213003e+11	1.0	880.656500	...	4284.0

1680909 rows × 24 columns

```
# Retrieve column names and print
df.columns
```

```
Index(['home_market_value', 'city', 'college_degree', 'good_credit',
       'street_address', 'state', 'latitude', 'individual_id',
       'length_of_residence', 'curr_ann_amt', 'address_id', 'income', 'churn',
       'longitude', 'days_tenure', 'acct_suspd_date', 'has_children',
       'home_owner', 'county', 'cust_orig_date', 'marital_status',
       'date_of_birth', 'social_security_number', 'age_in_years'],
      dtype='object')
```

```
# Display information about the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1680909 entries, 0 to 1680908
Data columns (total 24 columns):
 #   Column                  Non-Null Count    Dtype
---  ------                  --------------    -----
 0   home_market_value       1588623 non-null  object
 1   city                    1668842 non-null  object
 2   college_degree          1680909 non-null  float64
 3   good_credit             1680909 non-null  float64
 4   street_address          1680909 non-null  object
 5   state                   1680909 non-null  object
 6   latitude                1427190 non-null  float64
 7   individual_id           1680909 non-null  float64
 8   length_of_residence     1680909 non-null  float64
 9   curr_ann_amt            1680909 non-null  float64
 10  address_id              1680909 non-null  float64
 11  income                  1680909 non-null  float64
 12  churn                   1680909 non-null  int64
 13  longitude               1427190 non-null  float64
 14  days_tenure             1680909 non-null  float64
 15  acct_suspd_date         193456 non-null   object
 16  has_children            1680909 non-null  float64
 17  home_owner              1680909 non-null  float64
 18  county                  1668842 non-null  object
 19  cust_orig_date          1680909 non-null  object

 20  marital_status          1680909 non-null  object
 21  date_of_birth           1680909 non-null  object
 22  social_security_number  1680909 non-null  object
 23  age_in_years            1680909 non-null  int64
dtypes: float64(12), int64(2), object(10)
memory usage: 307.8+ MB
```

```
# Check for missing values in each column
df.isnull().sum()
```

```
home_market_value          92286
city                       12067
college_degree                 0
good_credit                    0
street_address                 0
state                          0
latitude                  253719
individual_id                  0
length_of_residence            0
curr_ann_amt                   0
address_id                     0
income                         0
churn                          0
longitude                 253719
days_tenure                    0
acct_suspd_date          1487453
has_children                   0
home_owner                     0
county                     12067
cust_orig_date                 0
marital_status                 0
date_of_birth                  0
social_security_number         0
age_in_years                   0
dtype: int64
```

```python
# Count the number of unique values in the 'churn' column
df['churn'].value_counts()
```

```
churn
0    1487453
1     193456
Name: count, dtype: int64
```

```python
# Encode the 'marital_status' column using LabelEncoder
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
df['marital_status'] = le.fit_transform(df['marital_status'])
```

```python
# Define the features to be used for prediction
feat_cols = ['curr_ann_amt', 'days_tenure', 'age_in_years', 'income', 'has_children',
       'length_of_residence', 'marital_status', 'home_owner', 'college_degree', 'good_credit']

# Extract features (X) and target variable (Y)
X = df[feat_cols]  # Features
Y = df['churn']     # Target variable
```

```python
# Import train_test_split from sklearn.model_selection
from sklearn.model_selection import train_test_split

# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.33, random_state=42)
```