Credit Card Default Prediction

DTSA 5510: Unsupervised Algorithms in Machine Learning

Agenda

- Introduction - Problem Statement
- Overview of Dataset & EDA
- Data Distribution, Outliers, and Missing Data
- Model Setup
- Results!
- Conclusion - Lessons Learned

Problem Statement

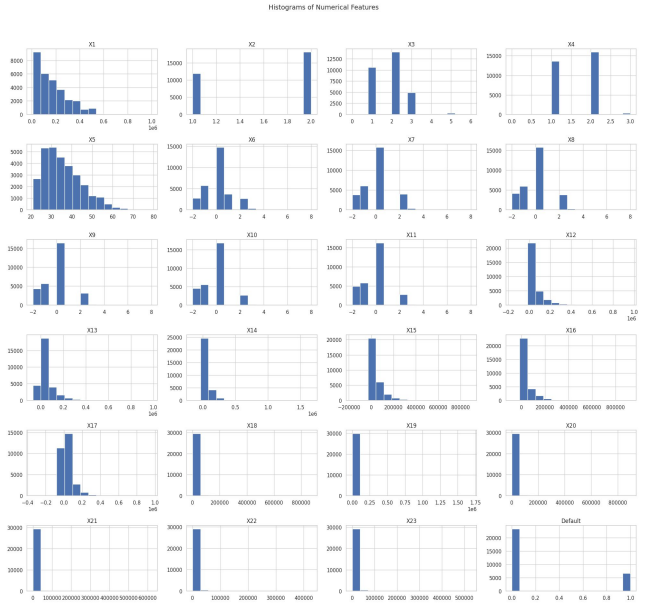
- Credit card default prediction is significant for financial institutions because it impacts their profitability and risk management strategies. Accurate prediction helps these institutions take proactive measures to mitigate potential losses.
- The primary aim of this project is to assess the efficacy of unsupervised learning techniques in predicting credit card defaults and compare their performance with traditional supervised learning methods.

Dataset Overview

1. **Dataset Overview**:
   - Title: "Default of Credit Card Clients" by Markelle Kelly, Rachel Longjohn, Kolby Nottingham hosted on UCI.
   - Size: 30000 client records and 23 distinct features.
2. **Data Composition**:
   - Includes financial, client payment history, and demographic data of clients
   - Features range from continuous numeric (credit card limit, payment history, bill amounts) to categorical (e.g., age, gender, education).
3. **Critical Variables**:
   - Outcome variable: `“Default”` (0 or 1 indicating default status).

Data Distributions

- The majority of the features (X1, X12-X23) exhibit right-skewed distributions, with most values concentrated towards the lower end of the range, indicating a common pattern in the data.
- Several features (X2, X4, X6-X11) show multimodal distributions, suggesting the presence of multiple subgroups or clusters within the data.
- The target variable, "Default," is binary and displays class imbalance, with more non-defaulters (0s) than defaulters (1s).



Missing Data

- No missing data noted in this data set.
- However, after data distribution modifications (de-skewing features), there were rows that had to be dropped due to np.inf and -np.inf values.
- Resulting in ~2000 rows being dropped.

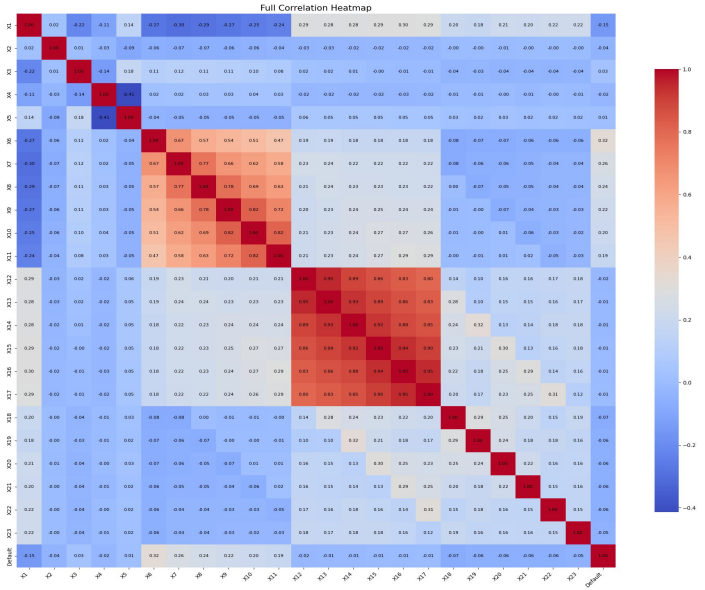
```
###Log transform for right-skewed features
right_skewed_features = ['X1', 'X12', 'X13', 'X14', 'X15', 'X16', 'X17', 'X18', 'X19', 'X20', 'X21', 'X22', 'X23']
for feature in right_skewed_features:
    df[feature] = np.log1p(df[feature])

###Power transform for left-skewed features
left_skewed_features = ['X3']
for feature in left_skewed_features:
    df[feature] = np.cbrt(df[feature])

###Replace -inf and inf values with NaN and then drop them
df.replace([np.inf, -np.inf], np.nan, inplace=True)
df.dropna(inplace=True)
```

Outliers

1. **Outlier Detection Strategy**:
    - Utilized z-score analysis with a threshold of 3 to identify outlier data points defining outliers as those significantly deviating from the mean in terms of standard deviation units.
2. **Flexibility and Adjustment**:
    - The method allows for adjustments in the sensitivity of outlier detection providing a robust tool for preprocessing and ensuring data quality by addressing extreme variations.
3. **Impact on Dataset**:
    - This approach led to the removal of a few hundred observations categorized as outliers refining the dataset for more representative and unbiased statistical analysis.

Data Correlation Matrix

1. **Significant Correlations and Redundancies**:
   - Detected strong correlations (>0.9) among variables indicating dependencies that could impact model performance due to multicollinearity and overfitting.
2. **Implications for Modeling and Data Reduction**:
   - High correlations among similar measurements and predictive metrics like suggest redundancy; removing one from each correlated pair can streamline models and enhance efficiency without losing essential information.
3. **Data Reduction**
   - 5 features dropped due to correlations >0.9



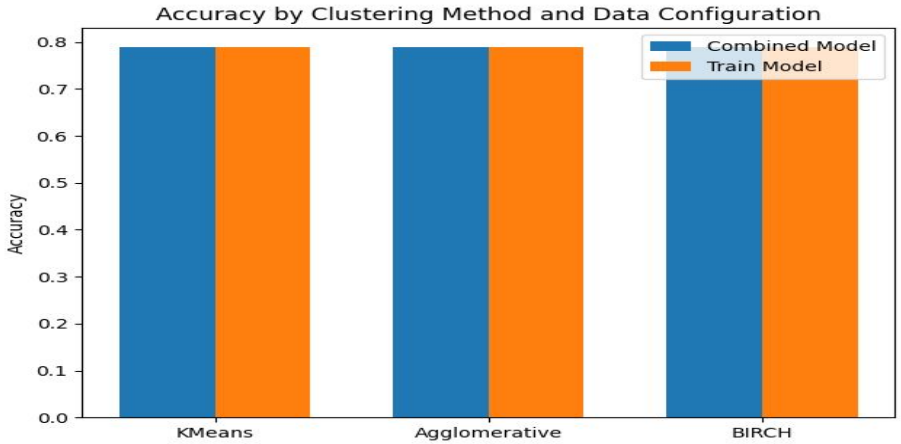
Model Training

1. **Preparation and Preprocessing**:
    - After preprocessing to standardize numerical features and one-hot encode categorical variables, dimensionality was reduced using PCA to focus on the most informative features enhancing model training efficiency.
2. **Model Training and Optimization**:
    - Trained a variety of models including logistic regression, random forest, XGBoost, optimizing hyperparameters via GridSearchCV to evaluate model effectiveness based on cross-validation accuracy.
    - For Unsupervised learning, the approach tested BIRCH, KMeans, and Agglomerative clustering techniques. Silhouette score was used to determine best model on validation set. Accuracy was determined applying majority class of a cluster to a test observation.
3. **Experimental Approaches**:
    - Conducted experiments where we performed data transformation, and then data transformation with SMOTE to handle class imbalance to compare impacts on model performance and generalizability.

Experiment Results - Unsupervised Learning w/ Data Transformations

Model	Configuration	Accuracy	Precision	Recall
KMeans	Combined	0.789853	0.394927	0.5
	Train	0.789853	0.394927	0.5
Agglomerative	Combined	0.789853	0.394927	0.5
	Train	0.789853	0.394927	0.5
BIRCH	Combined	0.789853	0.394927	0.5
	Train	0.789853	0.394927	0.5

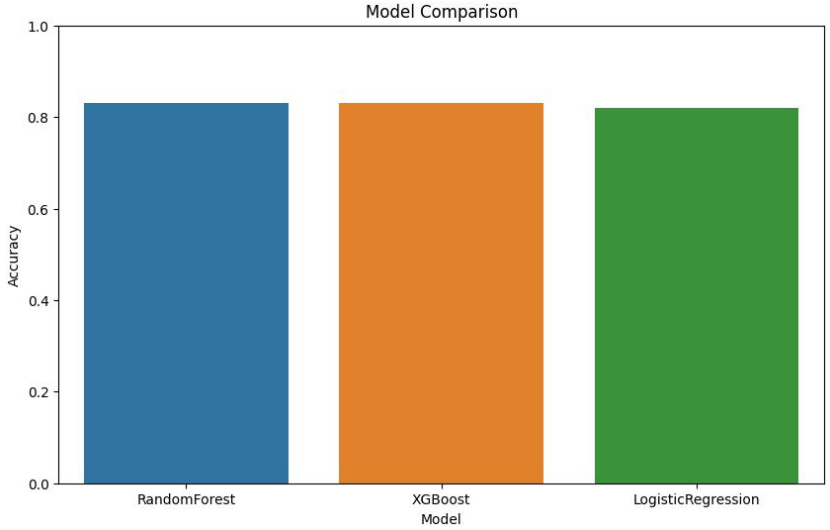
Experiment Results - Unsupervised Learning w/ Data Transformations



Experiment Results - Supervised Learning w/ Data Transformations

Model	Accuracy	Precision	Recall
XGBoost	0.8315086782376502	0.68	0.38
Random Forest	0.8317757009345794	0.68	0.38
Logistic Regression	0.822429906542056	0.64	0.33

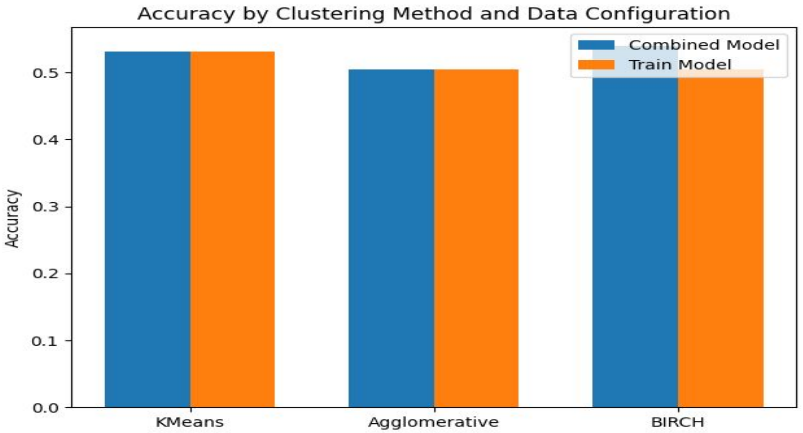
Experiment Results - Supervised Learning w/ Data Transformations



Experiment Results - Unsupervised Learning w/ Data Transformations + SMOTE

Model	Test Accuracy	Precision	Recall
KMeans (combined)	0.5307652974262826	0.542181178670545	0.5330312472035339
KMeans (train)	0.5305948525651951	0.5419826480105164	0.5328624423419539
Agglomerative (combined)	0.504857678540992	0.252428839270496	0.5
Agglomerative (train)	0.504857678540992	0.252428839270496	0.5
Birch (combined)	0.5399693199250043	0.551055280909285	0.5420175347865853
Birch (train)	0.504857678540992	0.252428839270496	0.5

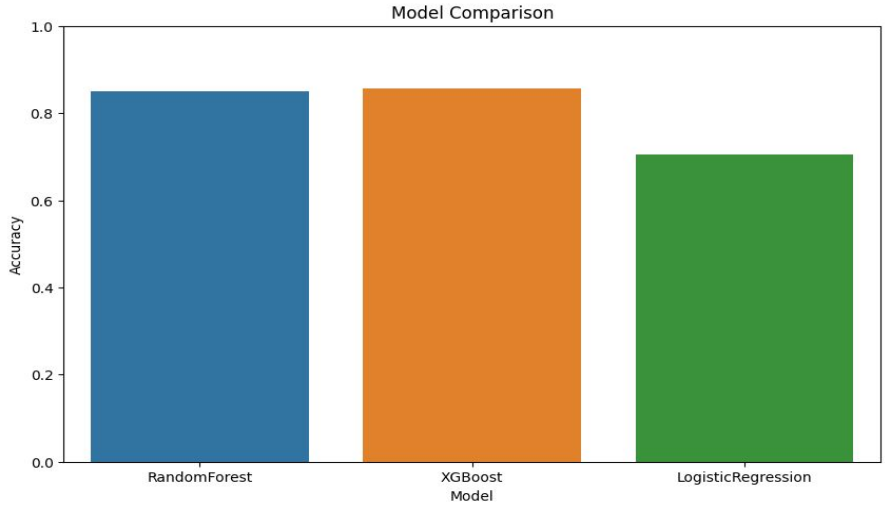
Experiment Results - Unsupervised Learning w/ Data Transformations + SMOTE



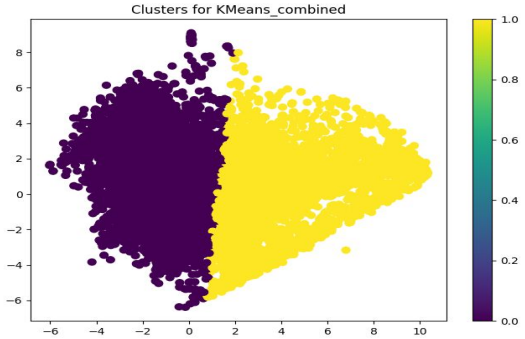
Experiment Results - Supervised Learning w/ Data Transformations + SMOTE

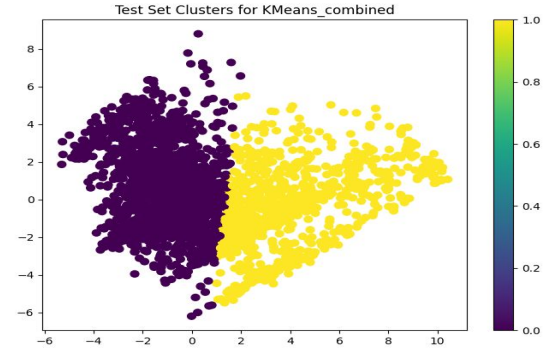
Model	Accuracy	Precision	Recall
XGBoost	0.85969967615476	0.85	0.87
Random Forest	0.84949718765979	0.85	0.85
Logistic Regression	0.70478950059655	0.74	0.62

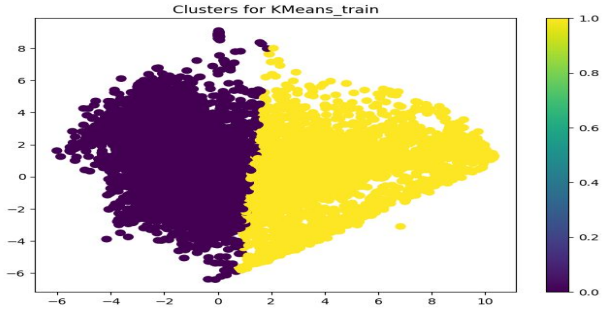
Experiment Results - Supervised Learning w/ Data Transformations + SMOTE

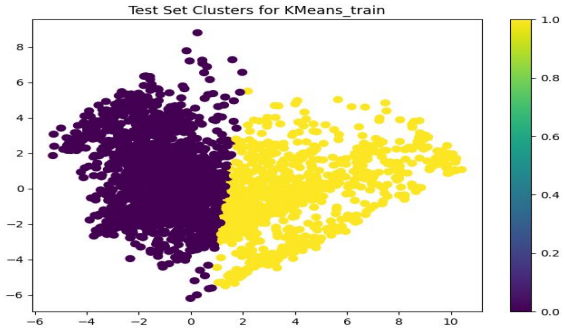


Cluster Visualizations









Conclusion - Lessons Learned

1. **Conclusion of Experiments**:
   - Despite employing strategies like GridSearch, dimensionality reduction, and SMOTE for class imbalance, the unsupervised models did not reach the performance standards necessary for production in a financial setting. However, supervised learning clearly excelled.
2. **Lessons Learned**:
   - The data doesn't contain enough structure to inform Unsupervised Techniques. Labels are very much needed.
   - Good clusters != Predictive performance
3. **Next Steps for Improvement**:
   - Explore additional Dimensionality reduction techniques
   - Better data collection - more observations might improve modeling effort.

Thank you!