# Data Collection and Preprocessing Phase

| | |
|---|---|
| Date | 10 July 2024 |
| Team ID | 740092 |
| Project Title | Credit card approval prediction using ML |
| Maximum Marks | 6 Marks |

**Data Exploration and Preprocessing Report**

Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modelling, and forming a strong foundation for insights and predictions.

| Section | Description |
|---|---|

Data Overview

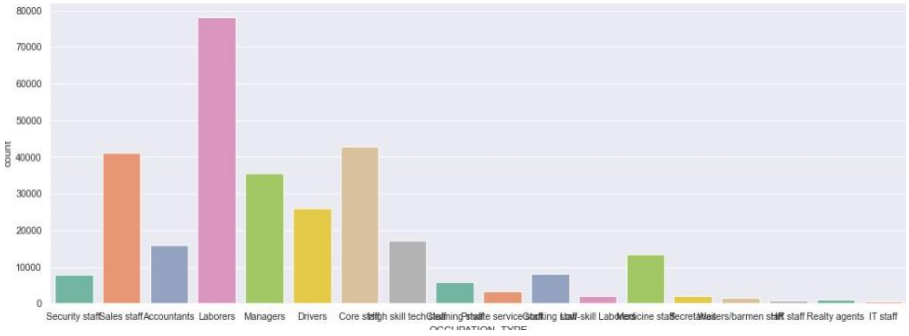
Dimension:

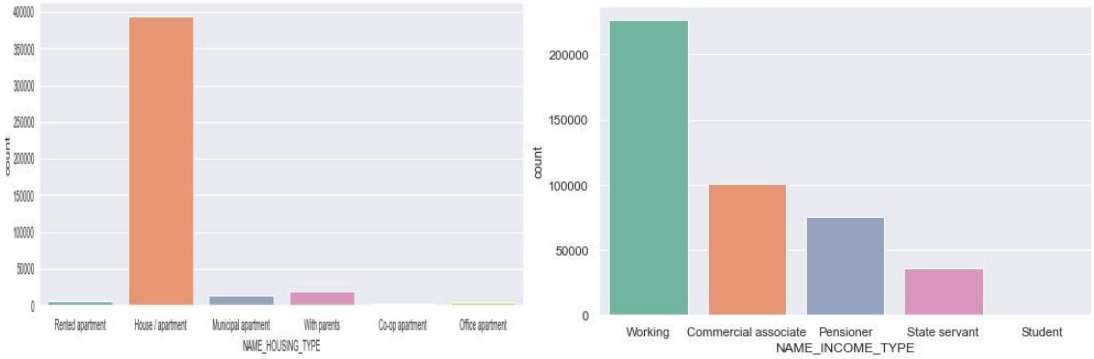
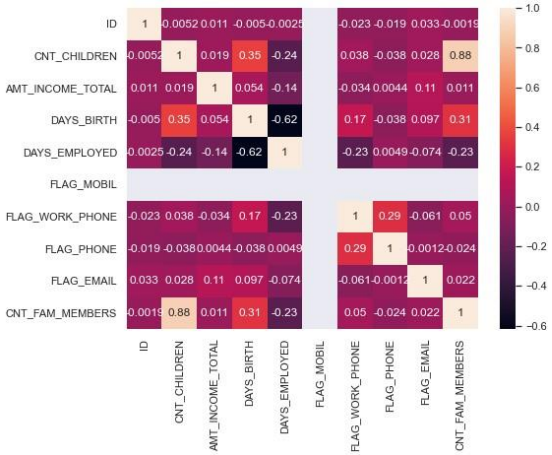
614 rows × 13 columns

Descriptive statistics:

| Feature | Count | Mean | Std | Min | 25% | 50% | 75% | Max |
|---|---|---|---|---|---|---|---|---|
| ApplicantIncome | 614 | 5403.46 | 6109.04 | 150 | 2877.50 | 3812.50 | 5795.00 | 81000 |
| CoapplicantIncome | 614 | 1621.25 | 2926.25 | 0 | 0.00 | 1186.50 | 2297.25 | 41667 |
| LoanAmount | 592 | 146.41 | 85.59 | 9 | 100.00 | 128.00 | 168.00 | 700 |
| Loan_Amount_Term | 600 | 342.00 | 65.12 | 12 | 360.00 | 360.00 | 360.00 | 480 |
| Credit_History | 564 | 0.842 | 0.365 | 0 | 1.00 | 1.00 | 1.00 | 1 |
| Age | 614 | 35.5 | 8.7 | 18 | 28.0 | 35.0 | 43.0 | 60 |
| Dependents | 614 | 0.5 | 0.7 | 0 | 0.0 | 0.0 | 1.0 | 3 |
| Approval_Status | 614 | 0.69 | 0.46 | 0 | 0.00 | 1.00 | 1.00 | 1 |

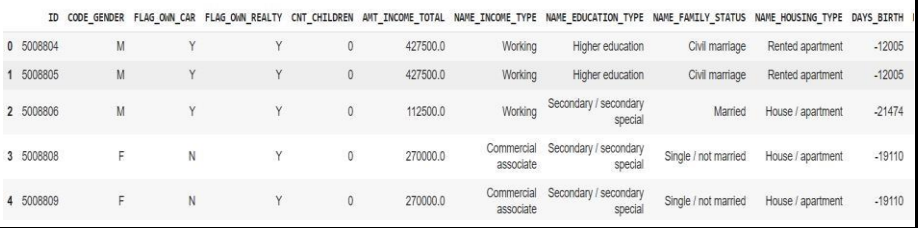
Univariate Analysis



| | |
|---|---|
| Bivariate Analysis |  |
| Multivariate Analysis |  |

| | |
|---|---|
| Outliers and Anomalies | - |

**Data Preprocessing Code Screenshots**

| | |
|---|---|
| Loading Data |  |
| Handling Missing Data |  |
| Data Transformation |  |
| Feature Engineering | Attached the codes in final submission. |
| Save Processed Data | - |

| | ID | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AMT_INCOME_TOTAL | NAME_INCOME_TYPE | NAME_EDUCATION_TYPE | NAME_FAMILY_STATUS | NAME_HOUSING_TYPE | DAYS_BIRTH |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5008804 | M | Y | Y | 0 | 427500.0 | Working | Higher education | Civil marriage | Rented apartment | -12005 |
| 1 | 5008805 | M | Y | Y | 0 | 427500.0 | Working | Higher education | Civil marriage | Rented apartment | -12005 |
| 2 | 5008806 | M | Y | Y | 0 | 112500.0 | Working | Secondary / secondary special | Married | House / apartment | -21474 |
| 3 | 5008808 | F | N | Y | 0 | 270000.0 | Commercial associate | Secondary / secondary special | Single / not married | House / apartment | -19110 |
| 4 | 5008809 | F | N | Y | 0 | 270000.0 | Commercial associate | Secondary / secondary special | Single / not married | House / apartment | -19110 |

```
data['Gender'] = data['Gender'].fillna(data['Gender'].mode()[0])
data['Marital_Status'] = data['Marital_Status'].fillna(data['Marital_Status'].mode()[0])

# Replacing + with space for filling the NaN values
data['Dependents'] = data['Dependents'].str.replace('+', '')

data['Dependents'] = data['Dependents'].fillna(data['Dependents'].mode()[0])

data['Dependents'] = data['Dependents'].fillna(data['Dependents'].mode()[0])

data['Self_Employed'] = data['Self_Employed'].fillna(data['Self_Employed'].mode()[0])

data['ApplicantIncome'] = data['ApplicantIncome'].fillna(data['ApplicantIncome'].mean())

data['LoanAmount'] = data['LoanAmount'].fillna(data['LoanAmount'].mean())

data['Credit_History'] = data['Credit_History'].fillna(data['Credit_History'].mode()[0])
```

```
data['Gender'] = data['Gender'].map({'Female': 1, 'Male': 0})
data['Married'] = data['Married'].map({'Yes': 1, 'No': 0})
data['Dependents'] = data['Dependents'].map({'0': 0, '1': 1, '2': 2, '3+': 3})
data['Education'] = data['Education'].map({'Graduate': 1, 'Not Graduate': 0})
data['Self_Employed'] = data['Self_Employed'].map({'Yes': 1, 'No': 0})
data['Property_Area'] = data['Property_Area'].map({'Urban': 2, 'Semiurban': 1, 'Rural': 0})
data['Loan_Status'] = data['Loan_Status'].map({'Y': 1, 'N': 0})

# Performing feature scaling using StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```