**Project Summary: Credit Approval Prediction**

**Objective**: To determine whether an individual should be granted credit based on their demographic and financial information.

**Steps and Process**:

1. **Data Acquisition and Cleaning**:
   * **Dataset**: I obtained a dataset containing information on customers such as age, credit limit, city, credit card product, company, and segment.
   * **Data Cleaning with SQL**:
     + I imported the dataset into a SQL database.
     + Cleaned the data by removing duplicates, handling missing values, and standardizing the format of numerical values (e.g., converting currency strings to numerical values).
     + Used SQL queries to explore and extract key insights from the dataset. For example, I identified the average credit limit across different cities and segments, the distribution of credit limits, and correlations between age and credit limit.
2. **Data Preprocessing for Machine Learning**:
   * Converted categorical variables into numerical values using one-hot encoding.
   * Created a target variable Credit\_Approved based on a threshold for the credit limit (e.g., credit limit > 100,000 INR is approved).
3. **Machine Learning Model Development**:
   * **Data Splitting**: Split the cleaned data into training and testing sets.
   * **Model Selection and Training**: Trained three different classification models:
     + **Random Forest Classifier**: An ensemble method that operates by constructing multiple decision trees.
     + **Logistic Regression**: A statistical model that uses a logistic function to model a binary dependent variable.
     + **K-Nearest Neighbors (KNN)**: A non-parametric method used for classification by comparing the closest training examples in the feature space.
4. **Model Evaluation**:
   * Evaluated each model using metrics such as accuracy, precision, recall, and F1 score.
   * Compared the performance of the models to determine the best one for predicting credit approval.
5. **Predictions**:
   * Made predictions on new examples using the trained models.
   * Validated the predictions to ensure the model's reliability in real-world scenarios.

**Results**:

* The Random Forest model provided the best performance with high accuracy and balanced precision and recall scores.
* Logistic Regression and KNN also performed well but were slightly less accurate compared to the Random Forest model.

**Conclusion**: Through this project, I successfully demonstrated the process of data cleaning using SQL and the application of machine learning techniques to predict credit approval. The insights derived from SQL queries helped in understanding the dataset better, and the machine learning models provided a robust solution for the credit approval prediction task.

The pd.get\_dummies function in pandas is used to convert categorical variables into a format that can be used in machine learning algorithms. Specifically, it performs one-hot encoding. Here's what the code you provided does:

python

Copy code

data\_encoded = pd.get\_dummies(data, columns=['City', 'Credit Card Product', 'Company', 'Segment'])

**What It Does**

1. **One-Hot Encoding**: For each specified categorical column, pd.get\_dummies creates new binary columns (one for each category) indicating the presence of that category. For example, if the City column has values 'New York' and 'Los Angeles', two new columns will be created: City\_New York and City\_Los Angeles. Each row will have a 1 in the column corresponding to its city and 0 in the other.
2. **Column Expansion**: The new DataFrame data\_encoded will have additional columns for each category in the specified categorical columns. The original categorical columns are dropped, and instead, binary columns representing each category are added.

Data Cleaning And Sanity Checks:  
  
**Age Data Handling**: For instances where customer age was recorded as less than 18, a meaningful treatment was provided. This likely involved either correcting erroneous data entries or excluding them from the analysis.

Solve:  
  
 **Impute with Median/Mean Age**: If you suspect these are errors and cannot correct them, you could impute the age with the median or mean age of the customer base. This is a simple approach when you don't have the exact data but want to maintain consistency.

 **Exclusion from Analysis**: If the number of records with age <18 is small and seems erroneous, you might exclude these records from the analysis entirely. This avoids introducing potential biases or inaccuracies into your results.  
  
import pandas as pd

# Load your dataset

df = pd.read\_excel('Credit\_Banking\_Project\_1.xls')

# Filter out records with age < 18

df\_cleaned = df[df['Age'] >= 18]

# Alternatively, if you want to impute the age with the median

median\_age = df['Age'].median()

df.loc[df['Age'] < 18, 'Age'] = median\_age **2. Credit Limit Violations: Identified any customers who had spent more than their credit limit in any particular month. This was crucial for understanding credit risk and customer behavior.**Filter out the records where the monthly spending exceeds the credit limit.  
import pandas as pd

datasetdf = pd.read\_excel('Credit\_Banking\_Project\_1.xls')# Assuming the

# 'CustomerID', 'Month', 'Monthly\_Spend', 'Credit\_Limit'

# Identify customers who exceeded their credit limit

df['Credit\_Violation'] = df['Monthly\_Spend'] > df['Credit\_Limit']

# Filter the rows where credit violation occurred

violations = df[df['Credit\_Violation']]

# Show the results

print(violations[['CustomerID', 'Month', 'Monthly\_Spend', 'Credit\_Limit']])  
  
import pandas as pd

# Load the Excel file

file\_path = '/mnt/data/Credit Banking\_Project - 1.xls'

xls = pd.ExcelFile(file\_path)

# Load the relevant sheet, assuming there's a sheet related to customer details

df = pd.read\_excel(xls, 'SheetName') # Replace 'SheetName' with the actual sheet name

# Convert the 'Age' column to integers

df['Age'] = df['Age'].astype(int)

# Save the DataFrame to a new Excel file if needed

df.to\_excel('/mnt/data/Credit\_Banking\_Project\_Converted.xlsx', index=False)  
  
**When dealing with missing values in the Age column, you can handle them using various techniques depending on the context and the nature of your data. Here are some common methods:**

**1. Imputation with a Statistic (Mean/Median/Mode)**

* **Mean Imputation**: Replace missing values with the mean age of the dataset.
* **Median Imputation**: Replace missing values with the median age (often preferred if the data is skewed).
* **Mode Imputation**: Replace missing values with the mode (the most frequent age).

**2. Forward or Backward Fill**

* **Forward Fill**: Fill missing values with the last observed value.
* **Backward Fill**: Fill missing values with the next observed value.

**3. Drop Missing Values**

* If the number of missing values is small, you might simply drop the rows containing missing ages.

# Option 1: Impute with the median age

median\_age = df['Age'].median() df['Age'].fillna(median\_age, inplace=True) # Convert the 'Age' column to integers df['Age'] = df['Age'].astype(int) #

Option 2: Impute with the mean age (alternative to median)

# mean\_age = df['Age'].mean()

# df['Age'].fillna(mean\_age, inplace=True)

# df['Age'] = df['Age'].astype(int)  
  
 **Forward Fill (propagate last valid observation forward)**:

* This method fills NaNs with the last valid value in the column.
* Example: df['Age'].fillna(method='ffill', inplace=True).

 **Backward Fill (propagate next valid observation backward)**:

* This method fills NaNs with the next valid value in the column.
* Example: df['Age'].fillna(method='bfill', inplace=True).

data = { 'Name': ['John', 'Jane', 'Jim', 'Jack'], 'Age': [28, None, 35, None], 'City': ['New York', 'Los Angeles', None, 'Chicago'] } df = pd.DataFrame(data) # Drop rows with any missing values df\_dropped = **df.dropna()** # Drop columns with any missing values df\_dropped\_cols = df.dropna(axis=1) # Drop rows with missing values in 'Age' or 'City' df\_dropped\_subset = df.dropna(subset=['Age', 'City'])  
  
**Insights Gained**

1. **High-Risk Customers**: Customers who exceeded their credit limits were identified, highlighting potential risks and the need for stricter credit control or personalized customer management strategies.
2. **Profitable Segments**: Certain customer segments were found to be more profitable, driven by higher spending or lower default rates. These segments are likely targets for future marketing or service enhancement.
3. **Age Group Spending Trends**: The analysis showed specific age groups that are more inclined to spend, which could guide targeted marketing campaigns or tailored financial products for those demographics.
4. **Spending Categories**: By understanding which categories attracted more spending, the bank could optimize its offerings, potentially developing new products or services that cater to these popular spending categories.