Credit Scoring Project Report

Since the data was already clean and scaled, I skipped these steps to move onto dealing with the categorical values. I made use of one hot encoding for certain values (those which could be dealt with as binary), and for others I used manual methods, such as using their averages.

Preprocessing:

- Removed all rows containing any NAN values (empty cells) to avoid issues with model training later.
- Removed the column using_pos which contained the same value for every single row ('No'). After confirming that the column contained no other values, I dropped it from the dataframe since it would have no effect on the training and prediction.
- Created a generalized function which dealt with all columns containing numerical data as strings, e.g., '101 to 200'. This function first extracted the number or numbers from the string and then analyzed whether to leave the number as it is, take its upper bound or calculate the average.
- Used LabelEncoder from sklearn library to encode 2 columns containing categorical data: shop_type and is_rental.

Data **before** any preprocessing:

number_of_orders	no_of_products	total_sales	gmv	avg_daily_sales	aprox_exist_inventory
1 to 50	101 to 200	258,923	200001 to 500000	6001 to 9000	100k - 300k

is_rental	shop_size	business_age(year)	electricity_bill	rent_amount	shop_type	using_pos	credit_score
Rented	272 - 408	9 to 12	More than 5000	0	General Store	No	4.1

Data **after** initial preprocessing:

number_of_orders	no_of_products	total_sales	gmv	avg_daily_sales	aprox_exist_inventory
25	150	258923	350000	7500	200000

is_rental	shop_size	business_age(year)	electricity_bill	rent_amount	shop_type	credit_score
2	340	10	5000	0	3	4.1

- Applied normalization using StandardScaler from sklearn, which utilizes Z-Score to center the data around the mean and scale it by the standard deviation. The actual formula used: *z = (x - mean) / std.* I applied normalization to the numerical data only, categorical remained as it is.

Resulting data:

number_of_orders	**no_of_products**	**total_sales**	**gmv**	**avg_daily_sales**	**aprox_exist_inventory**	**is_rental**
-0.46219	0.160205	-0.10893	0.071807	0.12428	0.256629	2
shop_size	**business_age(year)**	**electricity_bill**	**rent_amount**	**shop_type**	**credit_score**	
0.57671	0.648761	0.591991	-0.56571	3	4.1	

Data Splitting, Model Training and Prediction

- I used train_test_split from sklearn to split the data with a ratio of 80/20.
- I used the following models to train the data and make predictions:
    1. Linear Regression
    2. Random Forest
    3. XGBoost
    4. Neural Network
    5. Support Vector Machine
    6. KNN

Evaluating Accuracy

I used various evaluation metrics to understand the accuracies of the models:

1. Mean squared error,
2. Root mean squared error,
3. Mean absolute error,
4. R-squared,
5. Adjusted R-squared.

Findings and Recommendations

All the models performed well. Here are the models ranked by accuracy (according to the evaluation metrics used):

1. XGBoost
2. Random Forest
3. Neural Network
4. Support Vector Machine (SVM)
5. Linear Regression
6. K-Nearest Neighbours (KNN)

Here's the final table of values generated for all the models:

Model	MSE	RMSE	MAE	R-2	Adjusted R-2
Linear Regression	0.031099	0.176350	0.116912	0.948361	0.947126
Random Forest	0.027475	0.165755	0.109346	0.952060	0.950914
XGBoost	0.019979	0.141348	0.080132	0.967892	0.967124
Neural Network	0.023524	0.153375	0.090450	0.956995	0.955967
Support Vector Machine	0.023685	0.153899	0.082706	0.964893	0.964054
K-Nearest Neighbors	0.052190	0.228452	0.163845	0.915372	0.913349

While doing initial research for this project, I found logistic regression to be a widely popular model for credit scoring systems. However, when I tried using it, I faced issues. I found the mistake I was making was with misunderstanding of the data. Since our predicted class is continuous, logistic regression cannot be utilized. logistic regression explicitly deals with binary and multi class problems only.

Initially I included F1-Score as well, in the evaluation metrics. However, as can be seen by the table below, F1-Score is useless with regression tasks. Since our data is continuous, the F1-Score cannot be correctly calculated. Hence, I removed this from the list of metrics.

MODEL	F1-SCORE
XGBoost	1.0
Random Forest	1.0
Neural Network	1.0
Support Vector Machine (SVM)	1.0
Linear Regression	1.0
K-Nearest Neighbours (KNN)	1.0