Practical Data Mining Assignment-1

Name: Pratham Radhakrishna

ID NUMBER:s3997064

Part 1: Classification

1.

Run No	Classifier	Parameters	Training Error (%)	Cross-Validation Error (%)	Overfitting
1	ZeroR	None	30.0	30.0	None
2	OneR	-B 6	25.7	33.9	Moderate overfitting
3	J48	-C 0.25 -M 2	14.5	29.5	Moderate overfitting
4	IBK	-K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A "weka.core.EuclideanDistance -R first-last""	0.0	28.0	High overfitting

The results show that more complex models (J48 and IBK) tend to overfit, with training errors dropping to near zero but showing higher errors during cross-validation. This suggests that these models may be too specific to the training data and not generalizing well to new instances.While the more complex models (J48 and IBK) show lower training errors, their cross-validation performance does not reflect significant improvements. In contrast, simpler models like ZeroR and OneR are more stable but less effective overall.

2.

Run No	C	M	Training Error (%)	Cross-Validation Error (%)	Overfitting
1	0.05	2	24	30	Moderate overfitting
2	0.1	2	22.4	29.5	Moderate overfitting
3	0.15	2	17	29.6	Low overfitting
4	0.2	2	15.4	29.3	Low overfitting
5	0.25	2	14.5	29.5	Low overfitting
6	0.05	4	24	30.6	Moderate overfitting
7	0.1	4	24	29.3	Moderate overfitting
8	0.15	4	20.3	28.8	Low overfitting
9	0.2	4	17.7	28.6	Low overfitting
10	0.25	4	17.7	29.9	Low overfitting

Best Five Runs: After updating the values, the best five runs with the smallest difference between training error and cross-validation error are:

Run No	C	M	Training Error (%)	Cross-Validation Error (%)	Overfitting
9	0.2	4	17.7	28.6	Low overfitting
10	0.25	4	17.7	29.9	Low overfitting
4	0.2	2	15.4	29.3	Low overfitting
5	0.25	2	14.5	29.5	Low overfitting
8	0.15	4	20.3	28.8	Low overfitting

The analysis using the J48 classifier to minimise overfitting revealed that the best parameter combinations are C = 0.2 and M = 4, and C = 0.25 and M = 4. These settings showed the smallest difference between training error and cross-validation error, indicating low overfitting. Specifically, the run with C = 0.2 and M = 4 achieved a training error of 17.7% and a cross-validation error of 28.6%, demonstrating a good balance between training accuracy and generalisation performance. This suggests that these parameter values effectively reduce overfitting while maintaining robust predictive accuracy.

3.

Run No	Training Set Size (%)	Classifier Parameters	Training Error (%)	Cross-Validation Error (%)	Overfitting (Training - CV Error)
1	80%	Default J48	14.5	29.5	14.5 - 29.5 = -15.0
2	60%	Default J48	14.5	29.5	14.5 - 29.5 = -15.0
3	40%	Default J48	14.5	29.5	14.5 - 29.5 = -15.0
4	20%	Default J48	14.5	29.5	14.5 - 29.5 = -15.0

Conclusion: No matter how much data is utilised, the study shows that the training error is constant at 14.5% for all sizes of training sets, suggesting consistent model performance on the training set. Likewise, the cross-validation error is constantly 29.5%, indicating that different training data sizes have no effect on the model's capacity to generalise. Furthermore, the overfitting value stays constant at -15.0 for all sizes of training sets,

suggesting that the degree of overfitting is not affected by training set size. The steady overfitting value and consistency in training and cross-validation errors show that the model's performance is consistent and independent of the amount of training data.

4.

(k) Value	Accuracy	Precision	Recall	F1 Score	Overfitting Analysis
1	72%	0.716	0.720	0.718	High variance, possible overfitting
3	73.3%	0.718	0.733	0.721	Improved balance
5	74.2%	0.724	0.742	0.723	Good balance
7	74%	0.721	0.740	0.719	Good balance
9	74.3%	0.725	0.743	0.718	Optimal balance, minimal overfitting

- Smaller (k) values (like (k = 1)) can lead to higher variance and overfitting, as the model is more sensitive to noise in the training data.
- Larger (k) values (like (k = 9)) can reduce overfitting by averaging the predictions over more neighbours, but they might also lead to underfitting if (k) is too large.
- The performance metrics (accuracy, precision, recall, and F1 score) show improvement as (k) increases up to (k = 9).

5.

Classifier	Run	Parameter	Correctly Classified Instances (%)	Incorrectly Classified Instances (%)	Kappa Statistic	Mean Absolute Error	Root Mean Squared Error	Relative Absolute Error (%)	Root Relative Squared Error (%)
OneR	1	-B 6	66.1	33.9	0.0552	0.339	0.5822	80.6802	127.0545
OneR	2	-B 4	66.5	33.5	0.1119	0.335	0.5788	79.7282	126.3027
OneR	3	-B 2	64.2	35.8	0.0886	0.358	0.5983	85.2021	130.5665
IBk (k=1)	1	k=1	72	28	0.3243	0.2805	0.5286	66.7546	115.3422
IBk (k=4)	2	k=4	74.5	25.5	0.2698	0.3151	0.4338	74.9928	94.6713

The analysis of classifiers reveals that IBk with k=4 is the best in terms of predictive accuracy, achieving the highest correctly classified instances at 74.5%, indicating superior overall performance. Conversely, OneR (Run 3, -B 2) is the worst performer, with the lowest correctly classified instances at 64.2% and the highest mean absolute error and root mean squared error among the OneR runs. This suggests that OneR is less effective compared to IBk, particularly with k=4, which demonstrates better predictive accuracy and reliability.

---

6. Analysis and Conclusion

The analysis of classifiers ZeroR, OneR, and J48 reveals varying degrees of accuracy and predictive power. J48 emerges as the best-performing model with the highest accuracy at 70.5% and a substantial Kappa statistic of 0.2467, indicating significant predictive strength. In comparison, ZeroR, with an accuracy of 70.0% and a Kappa of 0, shows no predictive power beyond the majority class. OneR, with an accuracy of 66.1% and a minimal Kappa of 0.0552, has higher errors and worse accuracy, indicating its inability to capture data complexity as effectively as J48. Despite ZeroR and J48 having comparable accuracy, J48's greater Kappa and smaller errors (Mean Absolute Error and Root Mean Squared Error) make it a more reliable model. Therefore, J48 strikes the best balance between accuracy and predictive power among the three classifiers.

---

7. When the ZeroR, OneR, and J48 models are compared, the J48 model performs better than the majority class and has the greatest accuracy of 70.5%, as indicated by its Kappa Statistic of 0.2467. This indicates that the model has moderate predictive potential. Additionally, J48 exhibits reduced relative errors and lower error metrics, such as a Mean Absolute Error of 0.3467 and a Root Mean Squared Error of 0.4796, proving its accuracy and dependability. Even though OneR is less sophisticated, it still manages to attain a respectable 66.1% accuracy and a modest Kappa of 0.0552, demonstrating some predictive power. With a Kappa of 0 and an accuracy of 70.0% as a baseline, ZeroR highlights the significance of assessing models for more than simply accuracy and the necessity of significant predictive power. These findings highlight J48 as the best-performing model, balancing accuracy and predictive power effectively.

8.

Classifier accuracy significantly decreased when the attribute set was reduced using an attribute selection technique; it went from 70.5% on the whole dataset to 53% on the smaller dataset. The Relative Absolute Error and Root Relative Squared Error were higher, indicating that the reduced set did not capture the class distribution as effectively, even though the Mean Absolute Error and Root Mean Squared Error were lower on the reduced dataset, indicating closer average predictions to true values. In contrast to a Kappa of 0.2467 for the entire dataset, which demonstrated some agreement above random guessing, the Kappa score plummeted to 0 for the shortened dataset, suggesting that predictions were no more

accurate than guesswork. This analysis highlights that removing attributes reduced the model's overall accuracy, implying that the omitted attributes contained crucial information necessary for accurate classification.

Part 2: Numeric Prediction

1.

Classifier	Training Error (MAE)	Training Error (RMSE)	Cross-Validation Error (MAE)	Cross-Validation Error (RMSE)
ZeroR	39.3139	51.6914	39.4561	51.8398
M5P	36.7682	47.3153	39.8923	52.2994
IBk	0.7657	9.624	54.4554	71.7923

The analysis of classifiers ZeroR, M5P, and IBk reveals distinct performance characteristics. ZeroR, a baseline classifier predicting the mean or majority class, shows high training and cross-validation errors, serving as a benchmark for comparison. M5P, a model tree algorithm with linear regression models at the leaves, performs better than ZeroR, capturing relationships in the data with a reasonable balance between training and cross-validation errors, though there is slight overfitting. IBk, an instance-based learning algorithm (k-nearest neighbours), exhibits very low training error, indicating a strong fit to the training data, but significantly higher cross-validation error, suggesting poor generalisation and substantial overfitting. In conclusion, while ZeroR sets a baseline, M5P captures data patterns with minimal overfitting, and IBk, despite excellent training performance, fails to generalise well, indicating overfitting..

2.

Classifier	Parameters	Training Error (MAE)	Training Error (RMSE)	Cross-Validation Error (MAE)	Cross-Validation Error (RMSE)
M5P	-M 2	36.7682	47.3153	39.8923	52.2994
M5P	-M 4	36.7682	47.3153	39.8923	52.2994
M5P	-M 10	37.2748	48.5773	40.1076	52.6182
M5P	-R -M 4.0	37.5464	48.1989	40.4806	52.9426
M5P	-R -M 2.0	37.5464	48.1989	40.4806	52.9426
IBk	-K 1	0.7657	9.624	54.4554	71.7923
IBk	-K 3	31.3047	40.2861	44.7338	59.165
IBk	-K 5	34.3531	45.0498	42.8185	55.7985
IBk	-K 1 -W 1	73.3894	88.3849	49.9406	64.0715
IBk	-K 1 -W 2	51.264	66.3592	45.3663	59.8483

Best Five Runs

Here are the best five runs based on a balance between low training error, low cross-validation error, and low overfitting:

Classifier	Parameters	Training Error (MAE)	Training Error (RMSE)	Cross-Validation Error (MAE)	Cross-Validation Error (RMSE)
M5P	-M 2	36.7682	47.3153	39.8923	52.2994
M5P	-M 4	36.7682	47.3153	39.8923	52.2994
M5P	-M 10	37.2748	48.5773	40.1076	52.6182
IBk	-K 5	34.3531	45.0498	42.8185	55.7985
IBk	-K 3	31.3047	40.2861	44.7338	59.1650

The analysis reveals that M5P models with different -M values exhibit minimal overfitting and consistent performance across training and cross-validation, with the -M 2 and -M 4 configurations offering the best balance between low errors and stability. The IBk model with -K 5 demonstrates a good balance between predictive accuracy and generalisation, showing moderate training errors and relatively low cross-validation errors, making it a robust choice with less overfitting compared to -K 1. Although IBk -K 3 has slightly higher cross-validation error than IBk -K 5, it still provides good generalisation and ranks among the top performers. In conclusion, the M5P models (-M 2, -M 4, -M 10) and IBk models (-K 3, -K 5) are the top performers, with M5P models showing particularly strong consistency and low overfitting.

3

Run No	Classifier	Parameters	Training Error (MAE)	Training Error (RMSE)	Cross-Validation Error (MAE)	Cross-Validation Error (RMSE)	Overfitting
1	RF	-P 100 -I 100 -depth 10	17.6329	23.1047	39.7073	52.7153	Moderate overfitting
2	RF	-P 100 -I 100 -depth 20	15.0098	19.6214	39.629	52.5995	Moderate overfitting
3	RF	-P 100 -I 50 -depth 10	17.6311	23.2552	40.1018	53.2371	Moderate overfitting
4	RF	-P 100 -I 50 -depth 20	15.0245	19.5256	39.2747	52.1641	Moderate overfitting
5	RF	-P 100 -I 150 -depth 30	14.9575	19.8269	39.7355	52.6388	Moderate overfitting
6	SVR	-C 0.1 -I "PolyKernel -E 1.0"	35.6959	48.7329	39.9674	53.216	Moderate overfitting
7	SVR	-C 1.0 -I "PolyKernel -E 1.0"	35.6771	48.8732	39.9875	53.5246	Moderate overfitting
8	SVR	-C 1.0 -I "PolyKernel -E 3.0"	2.896	11.2853	95.6779	133.2661	High overfitting
9	SVR	-C 10.0 -I "PolyKernel -E 1.0"	35.6754	48.864	40.1688	53.6374	Moderate overfitting
10	SVR	-C 100.0 -I "PolyKernel -E 1.0"	35.6754	48.9229	40.2058	53.6371	Moderate overfitting

Table with Best Five Runs:

Run No	Classifier	Parameters	Training Error (MAE)	Training Error (RMSE)	Cross-Validation Error (MAE)	Cross-Validation Error (RMSE)	Overfitting
1	RF	-P 100 -I 100 -depth 10	17.6329	23.1047	39.7073	52.7153	Moderate overfitting
2	RF	-P 100 -I 100 -depth 20	15.0098	19.6214	39.629	52.5995	Moderate overfitting
3	RF	-P 100 -I 50 -depth 20	15.0245	19.5256	39.2747	52.1641	Moderate overfitting
4	RF	-P 100 -I 150 -depth 30	14.9575	19.8269	39.7355	52.6388	Moderate overfitting
5	SVR	-C 0.1 -I "PolyKernel -E 1.0"	35.6959	48.7329	39.9674	53.216	Moderate overfitting

The **Random Forest** classifier with `-P 100 -I 100 -depth 20` gives the best performance in terms of predictive accuracy and overfitting. It achieves the lowest cross-validation errors while maintaining moderate overfitting, making it the most reliable model for numeric prediction in this scenario.

4. Several important conclusions were drawn from the analysis of different classifiers for numerical predictions. The Random Forest (RF) classifier, in particular when configured with the parameters -P 100 -I 100 -depth 20, continuously showed strong performance with the lowest cross-validation errors and moderate overfitting, underscoring its robustness for numerical prediction tasks. Higher K values (-K 3 and -K 5) offered superior generalisation, whereas the IBk classifier with -K 1 shown notable overfitting. The M5P model tree technique demonstrated consistent performance with varying parameter configurations, efficiently capturing data patterns without causing appreciable overfitting. As a baseline, ZeroR continuously displayed large mistakes, highlighting the necessity of more complex models like M5P and RF for improved forecast accuracy. The SVR model with -C 1.0 -I "PolyKernel -E 3.0" highlighted the risk of high overfitting, demonstrating the importance of careful parameter tuning. These insights guide the selection of appropriate models, with Random Forest and M5P standing out as reliable options for accurate and generalizable predictions.

Part 3: Clustering

1. We see that the within-cluster sum of squared errors (SSE) drops as K grows, indicating a better fit with more clusters, based on the K-means clustering findings for various values of K (1, 2, 3, 4, 5, 10, 20). On the other hand, the "elbow" point—when the SSE begins to

decline more slowly—is usually where the ideal number of clusters is determined. K=3 or K=4 seems to be the best value for this dataset as it strikes a decent mix between lowering SSE and preserving significant cluster sizes. The SSE is 112.988 with K=3 and 95.068 with K=4, both of which are much better than K=1 and K=2..As K increases further to 5, 10, and 20, the clusters become smaller and more specific, potentially capturing noise rather than meaningful patterns, with K=20 having the lowest SSE of 43.947 but likely overfitting the data. Therefore, K=3 or K=4 offers distinct and useful groupings for identifying different customer profiles, risk categories, or other relevant segments in the dataset, without the risk of overfitting associated with too many clusters. For a more precise determination of the optimal number of clusters, additional validation methods such as the silhouette score or cross-validation could be employed

---

2. Variations in the final clustering results can be caused by altering the initial centrocentric location in the K-means algorithm by changing the seed value. These variances include changes in cluster centroids, sizes, number of iterations to convergence, and within-cluster SSE. The noteworthy dissimilarities found between seeds 1, 10, 20, and 50 underscore the significance of executing the algorithm on several occasions using distinct seeds to guarantee resilience and consistency in the clustering outcomes. More dependable and consistent clustering may be obtained by averaging the results or choosing the optimal solution based on SSE and significant cluster attributes.

---

3. The Expectation-Maximization (EM) algorithm was run on a dataset with 1000 instances and 4 attributes (duration, credit amount, age, job), selecting 7 clusters based on cross-validation after 100 iterations, with a log likelihood of -17.06711. Cluster characteristics revealed distinct profiles: Cluster 0 had high credit amounts and older ages with mostly high-qualified/self-employed/management jobs; Cluster 1 featured moderate credit amounts and young ages with mostly skilled jobs; Cluster 2 included low credit amounts, very young ages, and a mix of unskilled resident and skilled jobs; Cluster 3 had moderate credit amounts and older ages with a similar mix; Cluster 4, the largest at 24%, had very low credit amounts and middle-aged individuals predominantly in unskilled resident jobs; Cluster 5 had very high credit amounts and middle-aged individuals with a mix of skilled and high-qualified jobs; and Cluster 6, with high credit amounts and young ages, mostly had skilled jobs. The clusters varied in size, with Cluster 0 being the smallest at 4%. The job distribution varied significantly across clusters, highlighting distinct employment profiles within each group.

4.

Number of Clusters

- **Before Normalisation**: 7 clusters
- **After Normalisation**: 7 clusters
- **Observation**: The number of clusters remained the same, indicating that the EM algorithm identified the same number of clusters in both normalized and non-normalized data.

The fact that both the number of clusters and their sizes were constant suggests that the data's fundamental structure was maintained. On the other hand, the normalised centroids and enhanced log likelihood indicate that the clusters are now more defined. The increased log likelihood indicates that the clustering results are much improved when the data is normalised before the EM algorithm is run. By guaranteeing that every characteristic is weighed equally, it prohibits any one attribute from controlling the clustering process. As a result, normalisation is an essential step in the pretreatment of data for clustering algorithms, leading to more accurate and stable clustering.

5.

- **minLogLikelihoodImprovementCV**: Changing this parameter did not affect the clustering results. The number of clusters, number of iterations, log likelihood, and cluster sizes remained the same.
- **minStdDev**: Increasing this parameter from 1.0E-6 to 1.0E-5 resulted in an increase in the number of clusters from 7 to 9 and an improvement in the log likelihood. Further increasing it to 1.0E-4 did not change the clustering results.
- **minLogLikelihoodImprovementIterating**: Changing this parameter did not affect the clustering results. The number of clusters, number of iterations, log likelihood, and cluster sizes remained the same.

The EM algorithm is sensitive to the `minStdDev` parameter, which can significantly affect the number of clusters and the log likelihood. Increasing the `minStdDev` allowed the algorithm to identify more distinct clusters within the data, resulting in a better fit. On the other hand, the `minLogLikelihoodImprovementCV` and `minLogLikelihoodImprovementIterating` parameters did not have a noticeable impact on the clustering results, suggesting that the algorithm's convergence criteria were already met with the default values.

---

6. Seven clusters represent the ideal division of the data, based on the output of the EM algorithm. Every cluster possesses unique attributes; among them, Cluster 4 comprises a substantial section of the dataset, comprising individuals with shorter loan terms, smaller credit amounts, and a preponderance of unskilled or skilled workers. Understanding various client profiles and creating financial goods or services that are tailored to their individual needs may be accomplished via the usage of this segmentation.

---

7. The EM (Expectation-Maximization) algorithm is probably a better option than K-means for the given dataset because it can handle clusters of various sizes and shapes, automatically determine the ideal number of clusters, and provide a probabilistic measure of cluster membership. These features make the EM algorithm more accurate and robust for deciphering the underlying structure of the data. Although K-means is easier to use, quicker, and more computationally efficient, making it a good choice for short, initial clustering jobs and big datasets, its flexibility is limited by the

need to define the number of clusters in advance and the assumption of spherical clusters. K-means is useful for its speed and simplicity when the number of clusters is known, however EM is advised for more accurate and flexible clustering..

---

8. Using the K-means and EM algorithms on the dataset produced a number of insightful findings: Compared to K-means, which necessitates the pre-specification of cluster numbers, the EM algorithm found 7 optimum clusters, offering a more precise and nuanced segmentation. Interestingly, Cluster 4, which made up 24% of the data, was defined by smaller credit amounts, shorter loan terms, and a preponderance of skilled or unskilled resident jobs. These characteristics provided actionable client profiles for customised financial solutions. Better handling of borderline instances and variable cluster boundaries were made possible by the probabilistic nature of EM. The log likelihood was much increased by normalisation, highlighting the significance of data pretreatment**.** Additionally, the sensitivity of the EM algorithm to the `minStdDev` parameter demonstrated the need for careful parameter tuning to achieve optimal results. These insights collectively enhance the understanding of the dataset's structure, guiding resource allocation and targeted interventions.

Part 4: Association Finding

1. The two files, `groceries1` and `groceries2`, represent shopping transactions differently. `groceries1` uses a denser format where each attribute can be either `{F, T}`, explicitly indicating whether an item is present (T) or not present (F) in a transaction. In contrast, `groceries2` employs a sparser format where each attribute only has a single value `{T}`, indicating the presence of an item, with the absence of an item not explicitly represented. This makes `groceries1` suitable for algorithms requiring a complete binary matrix of item presence/absence, while `groceries2` is more efficient for algorithms that handle sparse data, as it reduces redundancy by listing only present items.

2.

- Running the Apriori algorithm on the groceries1.arff dataset with a minimum support of 0.95 and a minimum confidence of 0.9 resulted in 7 attributes and 4917 instances. The significant rules found predominantly reflect the absence of certain items leading to the absence of baby food, with all rules having a confidence of 100% and a lift of 1. For example, rules like "cooking_chocolate=F ==> baby_food=F" indicate that transactions not including cooking chocolate also do not include baby food. While these rules have high confidence, they are not particularly insightful for understanding purchasing patterns as they do not indicate positive associations. The high support and confidence thresholds likely led to rules that reflect the absence of items rather than meaningful co-occurrence patterns, as indicated by the lift values of 1, showing no increased likelihood beyond chance. Lowering these thresholds might reveal more meaningful associations.
- Exploring different confidence values (0.9, 0.8, 0.7, 0.6) with a constant minimum support of 0.95 in the Apriori algorithm did not change the number of rules, cycles, or clusters, nor did it affect the best rules found or the log likelihood. The top rules consistently indicated that the absence of items such as cooking chocolate, canned fish, and cake bars led to the absence of baby food, all with a confidence of 1. This consistency suggests that the high minimum support threshold strongly influences the results, leading to rules that primarily reflect the absence of items rather than meaningful associations. Lowering the support threshold might reveal more insightful associations, as the current settings are too stringent to capture varied purchasing patterns.

3.

- Many of the rules have `whole_milk=T` as the consequent, indicating that whole milk is frequently bought together with various combinations of other items. The high lift value (3.91) suggests that the presence of items like flour, root vegetables, and whipped sour cream significantly increases the likelihood of whole milk being purchased. Another significant association is with `other_vegetables=T`. For instance, `citrus_fruit=T, root_vegetables=T, tropical_fruit=T, whipped_sour_cream=T ==> other_vegetables=T` with a lift of 5.17. This shows that customers who buy a combination of fruits and whipped sour cream are also likely to buy other vegetables.**Lift**: Lift indicates the strength of an association rule over the random co-occurrence of the items. A lift value greater than 1 implies a positive association between the antecedent and the consequent. **Confidence**: Confidence measures how often the items in the consequent appear in transactions that contain the antecedent. A confidence of 1 means the rule is always true for the transactions in the dataset. **Conviction**: Conviction compares the probability that the antecedent occurs without the consequent (1 - confidence) to the probability of the consequent. Higher conviction values indicate stronger rules.
- For all three confidence values (0.5, 0.7, and 0.9), the number of rules generated is the same (10). The example rule `flour=T root_vegetables=T whipped_sour_cream=T ==> whole_milk=T` appears consistently across all confidence thresholds. The confidence, lift, and conviction values for the example rule remain the same across different confidence thresholds. This consistency suggests that the top rules with high confidence (1.0) are robust and do not change with varying confidence thresholds in this particular dataset. The Apriori algorithm generates the same top 10 rules with high confidence (1.0) regardless of the confidence threshold set to 0.5, 0.7, or 0.9. These rules have a strong association, as indicated by the high lift and conviction values.

---

4. FP-Growth generated significantly more rules compared to Apriori. This might be due to the different underlying mechanisms of the two algorithms. FP-Growth is generally faster and can handle larger datasets more efficiently, leading to the discovery of more rules. Both

algorithms identified strong associations with `whole_milk=T` as the consequent, but the antecedents differ slightly. Both algorithms produced rules with high confidence and lift values, indicating strong associations. However, the conviction values differ, suggesting differences in the strength and reliability of the rules between the two algorithms. **Apriori**: Generates a smaller number of highly confident rules, which may be easier to interpret and analyse. **FP-Growth**: Generates a larger number of rules, offering more insights but potentially requiring more effort to interpret.

---

5. The analysis using Apriori and FP-Growth algorithms revealed consistent strong associations with whole milk, indicating it as a central item in many transactions. High-confidence rules like `flour, root vegetables, whipped sour cream ==> whole milk` and `sugar, rice ==> whole milk` suggest strategic opportunities for retailers to place these items together or create promotional bundles. These insights can enhance inventory management and store layout, ensuring associated items are well-stocked and easily accessible, thus boosting sales. Further analysis with the Filtered Associator could provide additional valuable insights

---

6. The Apriori algorithm applied to the credit-g dataset revealed several strong associations. Individuals with critical or existing credit histories, combined with specific purposes (e.g., used car, radio/TV) and job types (e.g., high qualified/self-employed/management), consistently lead to a good credit class. Additionally, certain credit histories (e.g., 'all paid', 'no credits/all paid') paired with the purpose of furniture/equipment are strongly associated with skilled jobs. These insights indicate that high-quality credit histories and specific job types are key indicators of a good credit class, valuable for credit risk assessment and targeted financial product offerings