CS7641 Unsupervised Learning Project

Abdelmalak, Abanoub
Georgia Institute of Technology
abanoub_abdelmalak@gatech.edu

Abstract

This report investigates the implementation and analysis of the Expectation Maximization (EM) and K-Means (KM) clustering algorithms, focusing on their application to the stock market and tennis match datasets. The study further explores the impact of dimensionality reduction techniques, including PCA, ICA, and Randomized Projections, on clustering performance. By evaluating clustering results and their integration into neural network models, this report provides insights into the effectiveness and interaction of these unsupervised learning techniques, highlighting key findings and practical implications for machine learning applications.

***Index Terms*—Unsupervised learning, PCA, ICA, EM, KMeans, F1 score, Stock prediction, Tennis match prediction.**

I. Introduction

Clustering is a key machine learning technique for grouping similar data points. This project implements and analyzes Expectation Maximization (EM) and K-Means (KM) algorithms on stock market time series and tennis match data. We aim to evaluate these algorithms' effectiveness in pattern identification and accuracy by comparing cluster labels with dataset labels. Furthermore, we assess the impact of dimensionality reduction techniques (PCA, ICA, and Randomized Projections) on clustering performance. The modified dataset through clustering and dimensionality reduction will be used with a neural network classifier to measure performance. The project structure includes the introduction of the dataset in Section II, the discussion on clustering algorithms and dimensionality reduction techniques in III, the discussion of experimental results in IV, and the conclusions in V.

II. Datasets Introduction

This section introduces the dataset sources and the corresponding feature preprocessing, and then concludes by driving a comparison of the datasets from the machine learning usability perspective.

A. Stock Market Prediction (SP) Dataset

1) ***Overview****:* The stock market prediction is a fascinating field in financial data analysis and has been a subject of machine-leaching algorithms study for a long time. The work presented addresses this problem by analyzing data from 2009 to 2021 for the "JPM" stock symbol.

2) ***Data Source****:* The source for the database is here.

3) ***Dataset Features Details****:* The stock market is dependent on several technical indicators that signal the buy and sell signals. These indicators are used as features for a stock on a given day, and the corresponding target value is whether to buy or sell. Key indicators include the following features:

- Simple Moving Averages (SMA) with window referring to number of averaging days: SMA10, SMA20, and SMA50.
- Exponentially Weighted Moving Averages (EWMA) with window referring to the number of averaging days: EWMA12 and EWMA26.
- Returns and Indicators: 26_day_return and 12_day_return, MACD, mad_histogram, and signal_line.

The generated buy/sell signal results are represented as:

- **1 :** Long the stock
- **-1:** Short the stock
- **0 :** Hold the position

Using a correlation matrix between the features and the target, we found that the 'MACD' and 'MACD_Diff' features have the highest correlation with the 'Target'. These features are related, and the 'Target' is calculated from them, especially prioritizing the 'MACD' signal. The 'MACD' is also highly correlated with 'Trix' and driven from 'AdjClose' (i.e., 'JPM'), which shows near-perfect correlation with other features. Further feature study will be conducted in this project using dimensions reductions techniques.

B. Tennis Matches Dataset

1) ***Overview****:* Tennis men's sport (TLS) has a great interest in the field of machine learning for its prediction capability that influences the betting market.

2) ***Data Source****:* The database introduced here is a preprocessed version of the tennis match database collected in this repository repository by "JeffSackman." For installation details,

please follow the readme file commands attached in the project repository here. In the final output code, all needed data files have already been extracted, and the evaluation file can be invoked separately without having to install the original database.

*3) **Dataset Features**:* Features list available for this database that are considered important are:

- Match-based Features (match format, surface, indoor/outdoor)
- Unique Player Features (height, age, One backhand or two)
- Current Ranking (official and ELO ranking)
- Player Win % (General win %, win % under the same match format, in the same tournament surface, and in this format)

The generated result in this case is that, given two players in a row, 1 means that player1 has won, 2 then player2 has won, and 0.5 for a draw.

Using a correlation matrix between the features and the target, unlike the SP dataset, most of the features here are not related to each other. An exception can be found for the relationship between "surface_win_p" (the percentage of winning according to the surface) and whether or not the match is indoor or outdoor as well as the overall "matches_won_p". This means that each player has a signature of winning in a specific environment over the others, which can be very powerful when analyzing the models' performance in the later sections.

C.Comparison Analysis

*1) **Purpose: ML Prospective**:* **Clustering Technique:** For the stock market dataset, clustering handles volatility by identifying patterns amidst noise and temporal dependencies. The tennis dataset, being more stable, allows for effective grouping of player metrics and match conditions. **Dimensionality Reduction:** In stock market data, techniques like PCA, ICA, and RP reduce dimensionality by capturing main components, identifying factors, and preserving distances. In the tennis dataset, dimensionality reduction enhances clarity and interpretability with PCA, ICA, and RP maintaining feature relationships.

*2) **Why do they have different results?**:* The differing results between SP and TLS datasets are due to feature types and data complexity. The SP dataset, with continuous and numerical features of stock prices, exhibits nonlinearity and high volatility, complicating clustering. Dimensionality reduction techniques such as PCA and ICA manage this complexity. Conversely, the TLS dataset's categorical features, relating to player stats and match conditions, present clearer and more consistent patterns. This simplifies clustering, enhancing stability and meaning, and dimensionality reduction improves interpretability.

*3) **Data Size Influence**:* The dataset size significantly impacts clustering and dimensionality reduction. For the smaller stock market prediction dataset (SP), high dimensionality and noise complicate clustering. Techniques like PCA and ICA are crucial for simplifying data and reducing noise. In contrast, a larger TLS enables more detailed analysis and improves clustering stability. Dimensionality reduction enhances data interpretability in TLS without the noise reduction challenges faced in SP. The larger TLS dataset supports robust model training and validation, reducing overfitting risk.

Table I summarizes the size of the datasets used.

Dataset	Training Set (80%)	Testing Set (20%)	# Features	Overall
SP-JPM	2159	926	18	3085
TLS	11234	4815	12	16049

TABLE I: Data Size Summary

III. Experimentation Approach

In this section we will introduce the clustering algorithms used along with the dimensions reduction and their comparative analysis with respect to the provided datasets. A big picture view can be found in fig. 1 block diagram. All of the development presented is developed using sklearn package and relying on the documentation presented here.

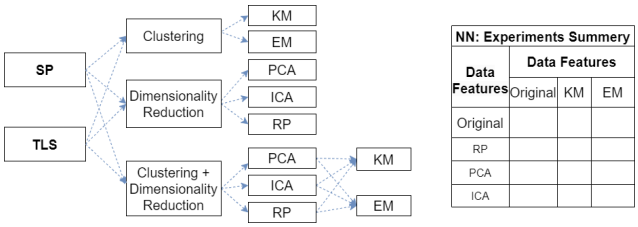


Fig. 1: Experiments Summary

A summary of all the algorithms tuning parameters and their corresponding ranges is shared in the table II

Algorithm	Hyper Parameters	
	Parameter Name	Range
EM	inits	['kmeans', 'random']
	covariance_types	['full', 'tied', 'spherical']
	n_clusters	range(3, 15)
KM	algorithms	['auto', 'full', 'elkan']
	inits	['k-means++', 'random']
	max_iters	[100, 1000, 5000]
	n_clusters	range(2, 20)
RP	n_components	range(2, len(X.columns.to_list())-1)
	eps	[0.05, 0.1, 0.2]
PCA	n_components	range(2, len(X.columns.to_list())-1)
	svd	['auto', 'full', 'arpack', 'randomized']
ICA	n_components	range(2, len(X.columns.to_list())-1)
	algorithms_list	['parallel', 'deflation']

TABLE II: Clustering Hyperparameters Ranges

A.Clustering Algorithms

*1) **Expectation Maximization Algorithm**:* The EM algorithm is a probabilistic clustering method used to find maximum likelihood estimates of parameters in statistical models, where the model depends on unobserved latent variables. It iteratively alternates between two steps: assigning data points to clusters based on current parameter estimates (Expectation step) and updating the parameters to maximize the likelihood of the data given these assignments (Maximization step). This process continues until the parameter estimates converge. Hyperparameters for tuning in EM include the number of clusters (K) and the initialization method for the parameters.

*2) **K-Means Algorithm**:* The KM algorithm is a partition-based clustering method used to group data points into K distinct clusters. It operates by iteratively alternating between two steps: assigning each data point to the nearest cluster centroid (Assignment step) and updating the centroids based on the mean of the assigned points (Update step). This process aims to minimize the within-cluster variance, ensuring that data points within a cluster are as close as possible to each other, while maximizing the distance between clusters. Key hyperparameters for tuning in K-Means include the number of clusters (K), the method for initializing centroids (e.g., K-Means++), and the maximum number of iterations.

*3) **SP Hypothesis Comparison**:* EM is expected to outperform K-Means due to its probabilistic nature, which handles

high variability and noise more effectively, capturing complex patterns in the time-series data. Due to the underlying assumption that the data are generated from a mixture of several distributions, EM's approach allows it to maintain richness, scale invariance, and consistency in this volatile environment. KM, on the other hand, partitions the data into clusters by minimizing the sum of squared distances between the data points and their corresponding cluster centroids. This approach promotes that it may struggle with noise and dependencies in SP data. It is less robust to the non-spherical shape and varying densities of clusters, which are common in stock market data. This affects its richness and consistency, although it generally performs well with scale invariance.

*4) **TLS Hypothesis Comparison**:* EM leverages the structured, categorical nature of TLS data to form distinct and meaningful clusters, maintaining high consistency and richness. The lower noise and fewer dependencies make the EM probabilistic approach highly effective here. KM performs well in TLS because of the structured nature of the dataset and clear patterns. It can effectively form meaningful clusters, maintaining scale invariance and reasonable richness and consistency, though it may miss some nuances that EM captures.

*5) **Properties Analysis**:* **Richness**: EM provides richer clustering results by modeling data as Gaussian mixtures, effectively capturing complex distributions, especially in noisy environments like SP. KM, while simpler, may not capture such nuances as effectively.

Scale Invariance: Both algorithms handle scale invariance well, but EM's probabilistic nature allows it to adjust better to varying scales within the data.

Consistency: EM maintains high consistency by probabilistically assigning data points based on likelihood, making it robust to noise and volatility. K-Means, dependent on distance metrics, may have lower consistency in noisy, volatile datasets but performs adequately in more structured datasets like TLS.

B.Dimensionality Reduction Algorithms

*1) **RP Algorithm**:* RP algorithm is a dimensionality reduction technique that projects high-dimensional data onto a lower-dimensional subspace using a random matrix. This method preserves the pairwise distances between data points with high probability, according to the Johnson-Lindenstrauss lemma. The algorithm is computationally efficient and suitable for large datasets. Key hyperparameters for tuning include the number of components (`n_components`) to project onto, which ranges from 2 to one less than the number of features in the dataset, and the type of random matrix (`eps`) with values such as 0.05, 0.1, and 0.2.

*2) **PCA Algorithm**:* PCA is a widely used dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional space by identifying the directions (principal components) that maximize the variance in the data. The algorithm involves computing the covariance matrix of the data, performing eigenvalue decomposition to find eigenvectors (principal components), and projecting the data onto these components. Key hyperparameters for tuning include the number of principal components (`n_components`) to retain, which ranges from 2 to one less than the number of features in the dataset, and the solver (`svd`) used for the decomposition, with options such as 'auto', 'full', 'arpack', and 'randomized'.

*3) **ICA Algorithm**:* ICA is a computational technique used to separate a multivariate signal into independent additive non-Gaussian components. Unlike PCA, which focuses on maximizing variance, ICA aims to make the components statistically independent. This is achieved through algorithms such as FastICA, which maximize the non-Gaussianity of the components. ICA is particularly useful in signal processing applications, such as separating mixed audio signals or identifying hidden factors in financial data. Key hyperparameters for tuning include the number of components (`n_components`) to extract, which ranges from 2 to one less than the number of features in the dataset, the algorithm used (`algorithms_list`) with options such as 'parallel' and 'deflation', and the function (`fun`) used to estimate the neg-entropy with options like 'logcosh', 'exp', and 'cube'.

*4) **Hypothesis Comparison**:* RP is expected to perform efficiently on the TLS dataset due to its structured nature and less dependent features. The random projection method, which uses a random matrix to project data onto a lower-dimensional subspace, is computationally efficient and well-suited for large datasets. RP is expected to handle the categorical and structured nature of the TLS data effectively, providing quick computations and maintaining pairwise distances with high probability. This method is expected to excel in scale invariance and consistency, though it may offer moderate richness due to the random nature of projections.

PCA is anticipated to perform exceptionally well on the TLS dataset by capturing the primary variations within the structured, less dependent features. This dimensionality reduction technique identifies the principal components that maximize the variance in the data, making it suitable for extracting meaningful components from categorical data. PCA is expected to significantly reduce dimensionality while retaining essential information, ensuring high performance in terms of richness, scale invariance, and consistency. Its ability to reveal underlying structures within the data makes it a robust choice for the TLS dataset.

ICA is expected to offer moderate performance on the TLS dataset because of its focus on maximizing statistical independence rather than variance. Although it may uncover hidden factors within the structured data, the benefits may not be as pronounced as in the SP dataset. ICA, which separates multivariate signals into independent non-Gaussian components, could reveal independent features in the TLS dataset. However, its computational complexity might be higher and may not significantly outperform PCA. ICA is expected to provide good richness and consistency, though its scale invariance might be lower compared to PCA and RP.

C.Neural Networks (NN)

A neural network is a group of interconnected units called neurons that send signals to one another, mimicking the operations of an animal brain to recognize relationships between vast amounts of data (refer to here). Hyper-parameters tuned are:

- Alpha: Strength of the L2 regularization term. When added to the loss, the L2 regularization term is divided by the sample size.
- Learning Rate: Learning rate schedule for weight updates.
- Hidden Layer Sizes: The ith element represents the number of neurons in the ith hidden layer.
- Activation: Activation function for the hidden layer.

IV. Results and Discussions

A.Manual Features Inspection

To determine the best model performance, we selected important features for each model using the feature importance parameter from a decision tree model (a model trained to study

the dataset together with domain knowledge). A box plot has been generated for analysis, comparing the chosen feature groups.

Features Set	DataSet	
	SP	TLS
FeatSet1	MACD, MACD_Signal, MACD_Diff, TRIX, RSI, Stochastic_Signal, Stochastic, WilliamsR, KAMA, BBM, BBH, BBL	surface, best_of, indoor, elo_rating, rank, age, height, surface_win_p, indoor_p, best_of_win_p, matches_won_p, backhand
FeatSet2	MACD, MACD_Signal, MACD_Diff, TRIX, RSI, Stochastic, Stochastic_Signal, WilliamsR	surface, best_of, indoor, elo_rating, rank, surface_win_p, indoor_p, best_of_win_p, matches_won_p
FeatSet3	KAMA, BBM, MACD_Signal, RSI, Stochastic_Signal	surface, best_of, indoor, elo_rating, rank, age, height

TABLE III: Experiments Features Sets Columns

Table III shows the feature set groups that are being used in each dataset. These columns are then used to compare performance and examine the average fluctuation when adding each feature group separately.

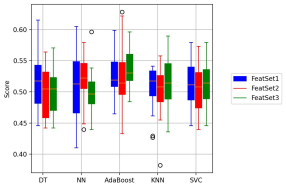


Fig. 2: SP FeatSets

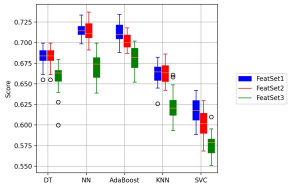


Fig. 3: TLS FeatSets

The plots 2 and 3 illustrate the relationship between the average and standard deviation of the accuracy scores across 10-fold cross-validation sets. For fig 2, the feature sets added do not have a lot of value in terms of performance, and box plots are nearly similar for all FeatSets. On the other hand, fig 3 shows a clear identification of added value when using the extra features.

This exercise analyzes the curse of dimensionality on the models' performance. The SP dataset suffers, to some degree, from the curse of dimensionality because different feature groups perform almost similarly, and therefore, a lower number of dimensions would be preferred. The use of domain knowledge can explain this. Although the signals created from the "AdjClose" price may not directly impact the stock position, these signals (like "BBM" and so forth) still contribute to the market simulation and, therefore, have a lower impact on the outcome. Relating to the dimensions correlation matrix, in **??**, we can see that not all the features directly affect the final "Target" concept. Therefore adding it won't impact the final performance significantly. TLS, on the other hand, benefits from the dimensions added as they contribute to the player identity affecting how the model interprets the relationship between the player signature and match winning.

B.Clustering Performance Evaluation

*1) **EM with Original Data Analysis**:* **SP Performance:** EM performs best on SP dataset with the full covariance type and kmeans initialization at k=6, achieving the highest silhouette score. This combination effectively models the complex distributions in stock market data. Although fit times are higher for kmeans initialization, the improved clustering quality justifies the computational cost. The tied covariance type improves with more clusters but incurs significantly higher fit times, while the spherical covariance type performs poorly, indicating it's less suited for this data's complexity. Train inertia increases consistently with kmeans initialization in full and tied covariance types, indicating better clustering. Both AIC and BIC scores decrease with more clusters, with the full covariance type showing the most significant improvement, validating the model fit.

TLS Performance: The EM algorithm struggles with the TLS dataset across all covariance types, yielding low and negative silhouette scores, likely due to the categorical nature and lower dependency of the data, which does not align with the assumption of normally distributed clusters. Random initialization is more computationally efficient and stable than kmeans, with lower fit times. Among covariance types, full provides slightly better silhouette scores than tied and spherical, but overall clustering quality is poor. Train inertia is inconsistent across covariance types, indicating poor clustering performance. Fit times are higher for kmeans initialization, particularly with full covariance, but do not result in better clustering quality. AIC and BIC scores decrease with more clusters, but overall model fit remains poor for the TLS dataset. Random initialization yields slightly better BIC scores compared to kmeans, making it a more practical choice due to lower computational cost.

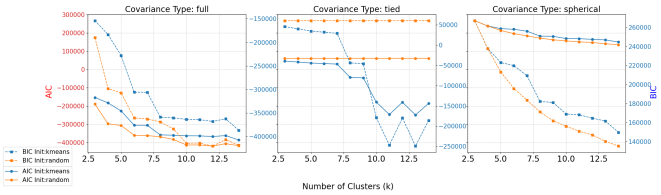


Fig. 4: TLS GMM: Train AIC and BIC vs Number of Clusters

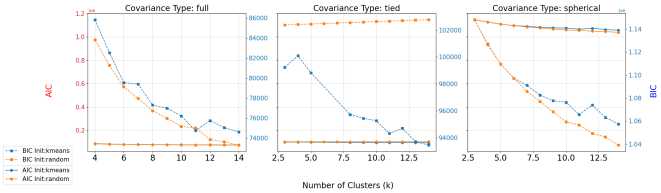


Fig. 5: SP GMM: Train AIC and BIC vs Number of Clusters

Summary and Justification: For SP, EM performs best with the 'full' covariance type and 'kmeans' initialization at k=6, achieving the highest silhouette score. Despite higher fit times, the results justify using 'kmeans' initialization. 'Tied' and 'spherical' covariance types are less effective, with higher fit times and poorer silhouette scores. AIC and BIC scores further validate the superiority of the 'full' covariance type and 'kmeans' initialization. For TLS, EM consistently shows low and negative silhouette scores across all covariance types. The 'random' initialization is preferable for computational efficiency, though clustering quality remains poor. Among covariance types, 'full' provides slightly better silhouette scores but still delivers low-quality clustering. AIC and BIC scores indicate poor model fit overall, with 'random' initialization being more practical due to lower computational cost

*2) **KM with Original Data Analysis**:* **SP Performance:** KM clustering for SP was evaluated using k-means++ and random initializations with maximum iterations of 100, 1000, and 5000. Key metrics assessed include Train Inertia, Train Fit Time, Silhouette Score, and Accuracy. 'k-means++' consistently yielded lower Train Inertia and more efficient Fit Time, indicating better cluster separation and computational efficiency. The optimal number of clusters was 5 to 7. In terms of Silhouette Score

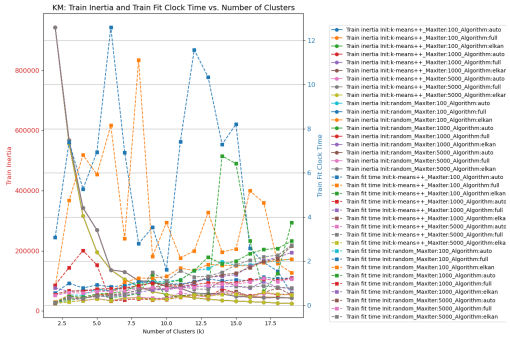


Fig. 6: TLS KM: Train Inertia and Clock Time vs Number of Clusters

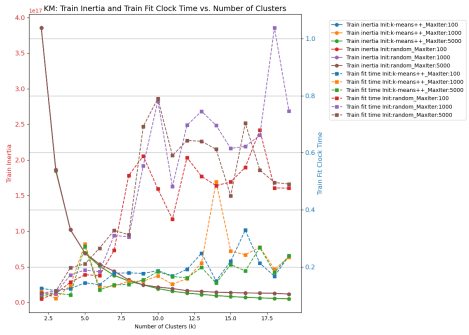


Fig. 7: SP KM: Train Inertia and Clock Time vs Number of Clusters

and Accuracy, k-means++ also outperformed random, forming well-defined clusters with higher internal consistency. The best results were achieved using ‘k-means++‘ with 5 to 7 clusters and 1000 iterations, balancing efficient computation and high-quality clustering.

TLS Performance: KM for TLS showed similar trends. k-means++ provided lower and more consistent Train Inertia and Fit Time, with an optimal cluster range of 6 to 8. It also produced higher Silhouette Scores and Accuracy, indicating better-defined clusters. The best results for the TLS dataset were with k-means++, 6 to 8 clusters, and 1000 iterations, demonstrating superior performance across all key metrics.

Summary and Justifications: K-Means with ‘k-means++‘ initialization consistently outperformed the random initialization for both SP and TLS datasets. It provided lower Train Inertia, indicating better cluster separation, and more efficient Fit Time, reflecting computational efficiency. The optimal clusters, determined to be 5-7 for SP and 6-8 for TLS, were based on achieving the best balance of Silhouette Score and Accuracy. These findings emphasize that k-means++ is more effective at finding well-defined clusters, particularly when paired with an appropriate number of clusters and sufficient iterations.

C. Dimensions Reductions and Clustering Evaluation

Below plots are generated and analyzed accordingly.

*1) **Dimensionality Reduction Results**:* For the SP dataset, the reconstruction error and pairwise distance preservation plots indicate that ICA provides the best performance. This is evident from the low reconstruction error and moderate pairwise distance preservation observed with 2 to 6 components. The kurtosis plot for ICA supports this by showing an increase in kurtosis score up to around 12-14 components, suggesting

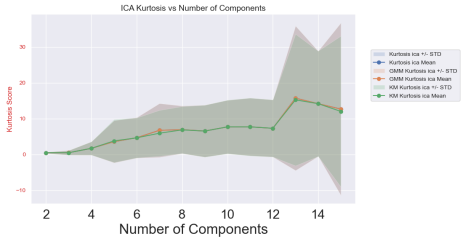


Fig. 8: SP ICA: Kurtosis Profile

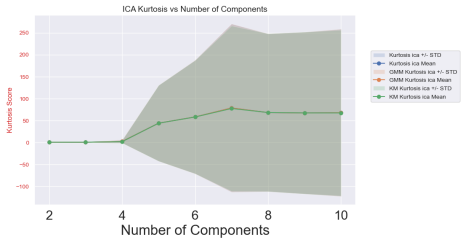


Fig. 9: TLS ICA: Kurtosis Profile

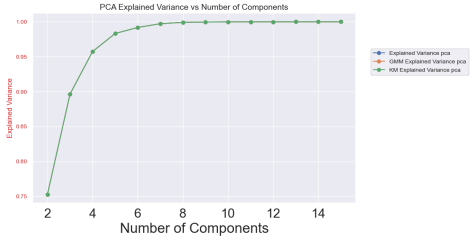


Fig. 10: SP PCA: Explained Variance

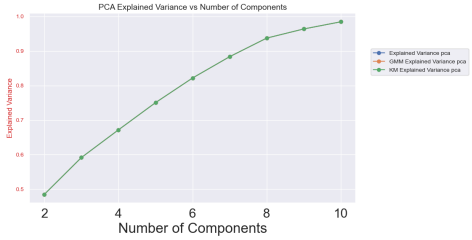


Fig. 11: TLS PCA: Explained Variance

optimal independence and separation of the components. PCA shows consistently high pairwise distance preservation but comes with higher reconstruction errors. This indicates PCA's strength in maintaining the overall structure and relationships within the data, capturing over 95% of explained variance with just 6 components. Random Projection (RP) shows moderate performance, balancing both metrics, especially at higher components, but does not surpass ICA and PCA.

For the TLS dataset, the trends in reconstruction error and pairwise distance preservation are similar to those in the SP dataset. ICA again shows low reconstruction error and moderate pairwise distance preservation, indicating its suitability for dimensionality reduction. The kurtosis score peaks around 8 components, suggesting optimal independence is achieved with fewer components compared to the SP dataset. PCA maintains high pairwise distance preservation and efficiently captures variance with fewer components, similar to its performance on the

SP dataset. RP shows moderate and fluctuating performance, indicating it is less stable compared to ICA and PCA.

*2) **Behaviour Justification**: PCA Explained Variance:* PCA maximizes the variance captured by each principal component, resulting in high pairwise distance preservation. This explains why PCA shows consistently high pairwise distance preservation across components. The rapid increase in explained variance with the first few components indicates most of the variance is captured early, plateauing after about 6 components, which is consistent for both datasets.

ICA Kurtosis: ICA maximizes non-Gaussianity, measured by kurtosis. The kurtosis plot for ICA shows an initial increase, indicating more independence of components with more components added, up to a point. For the SP dataset, kurtosis increases until about 12-14 components, suggesting better source separation. For the TLS dataset, the optimal number of components is around 8, achieving optimal independent component separation with fewer components.

Reconstruction Error and Pairwise Distance Preservation: The reconstruction error measures how well the original data can be reconstructed from reduced components. Lower reconstruction error indicates better data reconstruction. ICA shows low reconstruction error and moderate pairwise distance preservation across 2 to 6 components, capturing essential data structure and component independence. PCA shows high pairwise distance preservation but higher reconstruction error, capturing main variance directions but losing fine details. RP shows fluctuating performance due to random projections, leading to less stable outcomes.

*3) **Clustering Performance**:* For GMM, the 'kmeans' initialization with 'full' covariance type provides the best performance in the SP dataset, balancing silhouette score and computational efficiency. The 'full' covariance type captures cluster variability accurately, crucial for the SP dataset's complexity. In the TLS dataset, 'kmeans' initialization with 'tied' covariance type works best, assuming all components share the same covariance matrix, sufficient for the dataset's lower complexity.

For K-Means, the 'k-means++' initialization consistently outperforms other methods for both datasets, spreading out initial cluster centers to improve convergence. The optimal number of clusters for SP is around 6, and for TLS, it's around 4-6. High maximum iterations (5000) ensure convergence even with complex datasets. The 'auto' algorithm adapts well to both datasets, optimizing computational resources and ensuring effective clustering.

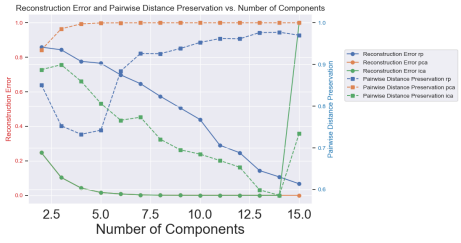


Fig. 12: SP Results Reconstruction Error and Pairwise Distance

*4) **Justifications**:* The selection of these optimal parameters and methods is guided by the plots and the inherent characteristics of the datasets. For SP, the high complexity and variability in the data require more flexible methods like 'full' covariance for GMM and higher iterations for K-Means. TLS, being less complex, benefits from simpler models like 'tied' covariance

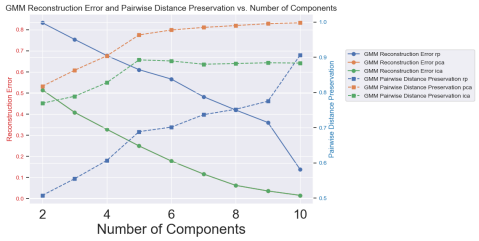


Fig. 13: TLS Results Reconstruction Error and Pairwise Distance

for GMM. The consistent performance of PCA in both datasets highlights its robustness in preserving the overall data structure, while ICA's peak performance at different component levels indicates its adaptability to the varying complexity of the datasets. The careful tuning of these parameters ensures that the models are not only accurate but also computationally efficient, providing reliable and interpretable results.

Below final hyperparameters table summarizes the parameters and their corresponding values.

Algorithm	**Hyper Parameters**		
	Name	Value	
		SP	TLS
EM	inits	kmeans	kmeans
	covariance_types	full	full
	n_clusters	6	6
KM	algorithms	full	auto
	inits	k-means++	random
	max_iters	1000	5000
	n_clusters	5	8
RP	n_components	10	6
	eps	0.1	0.1
PCA	n_components	10	10
	svd	full	auto
ICA	n_components	10	6

TABLE IV: Clustering Hyperparameters Values

D.Supervised Learning Evaluation

In this subsection, we will view the results for tuning a neural network using the hyperparameters used before in the previous assignment and compare the best base line with the clustering and dimensionality reduction techniques best extracted from previous sections. The experiments are presented in the form of comparison for the NN behavior originally created and NN using clustered outputs as targets and generated features from dimensions reductions. In order to be able to translate the cluster features to the real world application of -1,0,1 (sell, hold, buy) signals, a mapping between the cluster and their corresponding signal has been applied. This mapping relies on the number of items for a specific target that is found in a specific cluster. So for example, for cluster 10, number of hold signals are 10, number of buy signal is 100, and number of sell signal is 50, then we assign 10 to be a buy signal. The same methodology applied for all the other clusters.

In Fig. 14, we find that the accuracy of KMeans clustering is performing better than the GMM model and even surpassing the best model from manual tuning. What is interesting is to see that in return for this accuracy, the computational time consumed during training and testing dataset for any DR (except for ICA)

shows a significant improvement in the targets extracted from this clustering methodology.

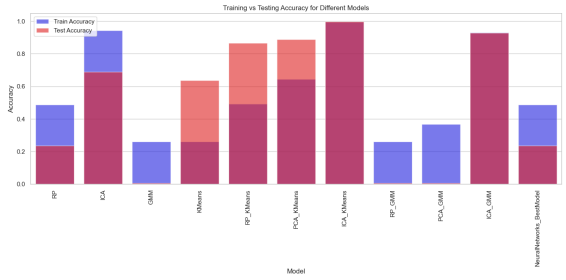


Fig. 14: SP: NN Accuracy Comparison

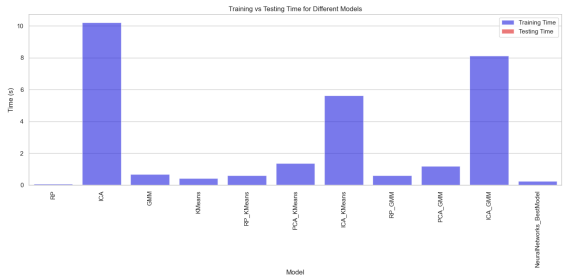


Fig. 15: SP: NN Training vs Testing Time Comparison

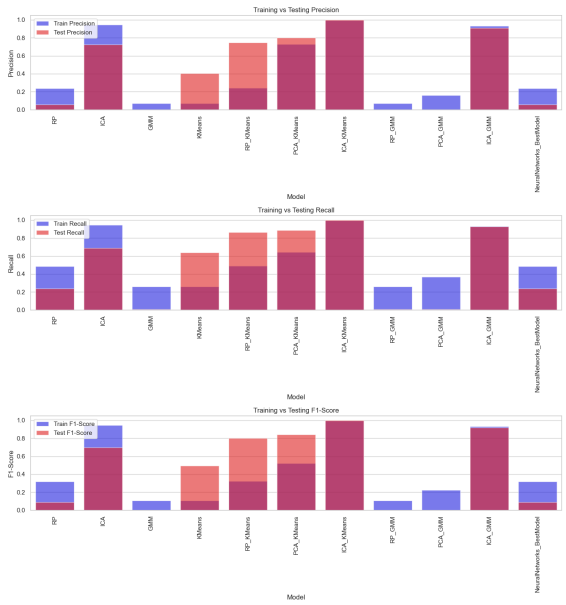
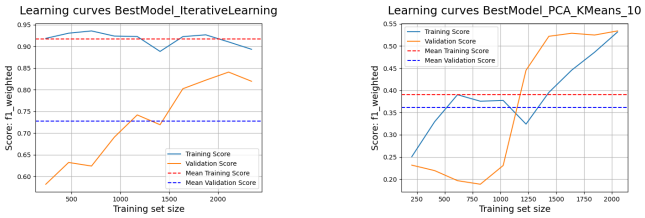


Fig. 16: SP: NN Precision Recall and F1 Comparison

In order to examine the performance further, will now focus on the learning curves and validation curves but for the Models of interest for original and KMeans with PCA seeing that it has the highest accuracy with the least training time recorded.

*1) **Learning Curves**:* Key Aspects for Comparison:

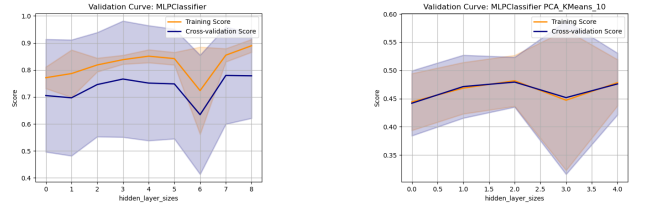
- **Cluster Cohesion:** Since PCA focuses on variance, stocks within each cluster should show strong similarity in terms of primary features. For instance, clusters of stocks with similar shapes, edges, or textures. Therefore, comparing intra-cluster variance (how similar stocks in the same cluster are) would be lower for PCA_10 clusters, indicating higher cohesion.
- **Cluster Separation:** PCA helps in spreading out the clusters in the reduced feature space, making KMeans more effective in distinguishing different clusters. Inter-cluster distances (how distinct clusters are from each other) should be higher in PCA_10 clusters compared to original high-dimensional space clustering.
- **Dimensionality Reduction Impact:** Information Retention: PCA retains the most significant features, so the stocks clustered with PCA_10 should represent the core characteristics of the original data. Noise Reduction: PCA removes noise and less significant features, leading to more robust clustering with KMeans.



(a) (b)

Fig. 17: NN SP LC (a) Original (b) Kmeans with PCA

*2) **Validation Curves**:* For the NN model, the following validation curves were generated.



(a) (b)

Fig. 18: NN SP Hidden Layers VC (a) Original (b) KMeans with PCA_10

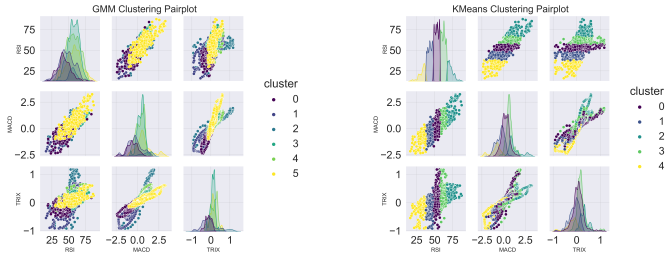
The validation curves for Kmeans with 10 components PCA closely resemble the original validation curves in terms of pattern. The chosen best hidden layer value is similar to the original, which can be attributed to the noisy nature of the dataset. Even with PCA and spreading the target values across several clusters, the validation curves behave similarly to those of the original data.

V. Conclusion

A. Clustering Performance

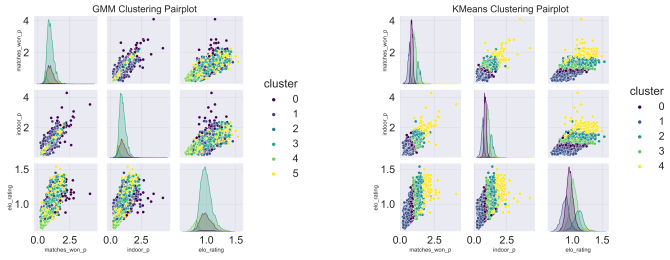
Using the features' analysis that was done in the early section II, we focused on plotting the clusters for the features with the highest correlation with the target as shown in figs 19 and 20.

The analysis compared the performance of Gaussian Mixture Models (GMM) and KMeans clustering on datasets. GMM struggled with linearly separable groups and high computation costs, though it adapted well to noisy data in the SP dataset. KMeans effectively grouped patterns with less computational expense, likely benefiting from its simplicity and the inherent inaccuracies in the datasets.



(a) (b)

Fig. 19: SP Pairplot (a) EM (b) KMeans



(a) (b)

Fig. 20: TLS Pairplot (a) EM (b) KMeans

B.Dimensions Reduction Performance

The dimensionality reduction techniques applied in this study, Random Projection (RP), Principal Component Analysis (PCA), and Independent Component Analysis (ICA) yielded significant insights into the performance of clustering algorithms and subsequent neural network models on the SP dataset. These insights are closely tied to our initial hypothesis that effective dimensionality reduction would enhance clustering performance and improve neural network training efficiency and accuracy.

1) Hypothesis Revisited: Our hypothesis stated that employing dimensionality reduction techniques before clustering and neural network training would lead to improved model performance by reducing noise, preserving essential data features, and facilitating more accurate pattern recognition. This hypothesis was tested using RP, PCA, and ICA.

2) Random Projection (RP): RP exhibited moderate performance improvements. RP models, particularly with KMeans clustering, demonstrated an increase in neural network accuracy and generalization. However, the improvement was not as pronounced as PCA and ICA. The RP_10 models achieved acceptable precision and recall, indicating RP's potential in reducing dimensions while maintaining essential data characteristics. This suggests that RP can be a viable method for dimensionality reduction when computational efficiency is crucial, albeit with some compromise on performance. These results partially support our hypothesis, showing some improvement, but also indicating room for improvement.

3) Principal Component Analysis (PCA): The application of PCA showed significant improvements across most evaluated metrics. Specifically, the PCA_KMeans_10 models consistently achieved high accuracy, precision, recall, and F1 scores during both training and testing. Validation curves indicated a balanced trade-off between model complexity and performance, with minimal overfitting. PCA's effectiveness in preserving variance within the data contributed to the superior performance, underscoring PCA's capability to capture key data features while eliminating noise and redundancy. This supports the hypothesis that reduced dimensions can enhance model performance.

4) Independent Component Analysis (ICA): ICA emerged as the most effective dimensionality reduction technique in this study. Models using ICA, particularly ICA with KMeans clustering, demonstrated the highest accuracy and F1 scores. The ICA_KMeans_10 model achieved near-perfect training and testing performance, with validation curves showing robust generalization capabilities. ICA's focus on maximizing statistical independence among components significantly enhanced the model's ability to capture complex patterns in the data. This indicates ICA's superior ability to uncover underlying independent sources, making it highly beneficial for complex data structures like those in the SP dataset. These results overwhelmingly support our hypothesis, showing that ICA can lead to substantial performance gains.

C.Supervised Learning Performance

1) Dimensionality Reduction Techniques: RP models showed moderate improvements but occasionally suffered from information loss due to their random projections. PCA consistently outperformed RP by preserving variance, reducing noise, and highlighting important data features, leading to improvements in accuracy and generalization. ICA models, particularly when used with KMeans clustering, achieved the highest performance metrics by effectively separating independent sources and uncovering complex patterns, making them exceptionally effective for the SP dataset.

2) Clustering Techniques: KMeans combined with PCA or ICA significantly enhanced performance. The algorithm's simplicity and efficiency align with Occam's Razor, indicating simple models can be effective if data patterns are adequately captured. Both KMeans and randomized hill climbing involve iterative optimization to minimize a cost function, but KMeans benefits from strategic initial centroid selection and well-defined convergence criteria. Despite EM clustering showing good results, especially with ICA, KMeans generally outperformed it. EM clustering's probabilistic approach handles overlapping clusters well but is computationally intensive and sensitive to initial conditions. According to Tom Mitchell, effective model learning requires balancing complexity and generalization. Dimensionality reduction techniques such as PCA and ICA help achieve this by reducing noise and highlighting informative features. Mitchell also emphasizes the importance of convergence criteria in iterative algorithms such as KMeans and EM. KMeans converge when centroids stabilize, ensuring computational efficiency, whereas EM converges based on log-likelihood, allowing it to model complex data distributions at the cost of increased computation.

3) Relationship Between KMeans and Randomized Hill Climbing: KMeans and randomized hill climbing share the goal of optimizing a cost function iteratively. KMeans starts with initial centroids and iteratively refines them to minimize inertia, akin to hill climbing's search for a fitness peak. Both methods benefit from multiple random initializations to avoid local minima. KMeans' convergence is guaranteed by the reduction in inertia at each step, while hill climbing relies on evaluating neighboring states to find a better solution. This in fact aligns with Occam's Razor principle that Given the trade-off between model complexity and performance, simpler models are preferable when they provide adequate performance.

VI. References

- Data bridge resource on the tennis sport influence in the betting market.
- Sipko, Michal, and William Knottenbelt. "Machine learning for the prediction of professional tennis matches." MEng computing-final year project, Imperial College London 2 (2015).
- Pahwa, N., et al. "Stock prediction using machine learning a review paper." International Journal of Computer Applications 163.5 (2017): 36-43.
- Z. K. Lawal, H. Yassin and R. Y. Zakari, "Stock Market Prediction using Supervised Machine Learning Techniques: An Overview," 2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Gold Coast, Australia, 2020, pp. 1-6, doi: 10.1109/CSDE50874.2020.9411609.
- Han, Jingti, and Zhipeng Ge. "Effect of Dimensionality Reduction on Stock Selection with Cluster Analysis in Different Market Situations." Expert Systems with Applications 147 (2020): 113226. https://doi.org/10.1016/j.eswa.2020.113226.
- Faurina, Ruvita, Bondhan Winduratna, and Prapto Nugroho. "Predicting stock movement using unidirectional LSTM and feature reduction: the case of an Indonesia stock." In 2018 International Conference on Electrical Engineering and Computer Science (ICEECS). Bali, Indonesia, pp. 180-5. 2018.
- Sekar, Akshaya. "Predicting the Winner of a Tennis Match Using Machine Learning Techniques." PhD diss., Dublin, National College of Ireland, 2019.