GROUP WORK PROJECT # 2
GROUP NUMBER: 4735

MScFE 632: Machine Learning in Finance

FULL LEGAL NAME	LOCATION (COUNTRY)	EMAIL ADDRESS	MARK X FOR ANY NON-CONTRIBUTING MEMBER
Marin Stoyanov	Bulgaria	azonealerts@gmx.com	
Guoying Li	United States	liguoying1019@gmail.com	
Ashutosh Kumar	India	skdubey.ashutosh@gmail.com	

Statement of integrity: By typing the names of all group members in the text boxes below, you confirm that the assignment submitted is original work produced by the group (excluding any non-contributing members identified with an "X" above).	
Team member 1	Marin Stoyanov
Team member 2	Guoying Li
Team member 3	Ashutosh Kumar

Use the box below to explain any attempts to reach out to a non-contributing member. Type (N/A) if all members contributed.
Note: You may be required to provide proof of your outreach to non-contributing members upon request.

N/A

LDA

● **Advantages:**

It is better interpretable than PCA.
It is a comparatively simple algorithm that is also computationally efficient.
It can handle multicollinearity.
It can even work with big datasets (such that have more features than the training samples) [1]

● **Basics:**

LDA is used primarily for dimensionality reduction like PCA but also can be used for classification. Unlike PCA, LDA increases interpretability and minimizes loss of information.

The goal of LDA is to project a dataset onto a lower-dimensional space with good class-separability in order to avoid overfitting ("curse of dimensionality") and also to reduce computational costs. It works by calculating the directions ("linear discriminants") that will represent the axes that maximize the separation between multiple classes. Like PCA it lowers the dimensions of the features and separates the features into different classes. [2]

● **Disadvantages:**

It assumes that the data has a Gaussian distribution
It assumes that the covariance matrices of the different classes are equal
It assumes that the data is linearly separable
It may not perform well in high − dimensional feature spaces.
It doesn't work well with outliers [1]

● **Equations:**

LDA uses the Fisher's discriminant that looks like this [3]:

$$S = \frac{\sigma^2_{between}}{\sigma^2_{within}} = \frac{(\vec{w} * \vec{\mu}_1 - \vec{w} * \vec{\mu}_0)^2}{\vec{w}^T * \Sigma_1 \vec{w} + \vec{w}^T * \Sigma_0 \vec{w}} = \frac{(\vec{w} - (\vec{\mu}_1 - \vec{\mu}_0))^2}{\vec{w}^T (\Sigma_1 + \Sigma_0) \vec{w}}$$

Where:

$\vec{\mu}_1$ *and* $\vec{\mu}_0$ *are the means for classes of observations* 1 *and* 0 *accordingly*

Σ_1 *and* Σ_o *are the covariances of class* 1 *and* 0 *accordingly*

$\vec{w}$ *is the vector of features*

After all the LDA assumptions are met, the maximum separations occurs when:

$$\vec{w} \propto (\Sigma_0 + \Sigma_1)^{-1} (\vec{\mu}_1 - \vec{\mu}_0)$$

● **Features:**
It doesn't work well when there are outliers because they affect the means of the classes and thus skew essential statistics like the mean and standard deviation, affecting the separation.
● **Guide:**
The input data should be normally distributed. The outliers should be removed or at least applied to a lognormal function. In order for all the data to have the same variance it should be standardized so that the assumption can be met. [2]

● **Hyperparameters:**
$solver: \{‘svd’, ‘lsqr’, ‘eigen’\}, default = ’svd’$
This is the hyperparameter that is responsible for the method of how the LDA is to be calculated. It's options include: Singular value decomposition (default), Least squares solution and Eigenvalue decomposition

$shrinkage: ‘auto’ \; or \; float, \; default = None$
These are the options for the shrinkage. The default is none and the rest options are automatic shrinkage using the Ledoit-Wolf lemma or float: fixed shrinkage parameter between 0 and 1

$priors: array - like \; of \; shape \; (n\_classes,), \; default = None$
This hyperparameter stands for the class prior probabilities which are inferred from the training data and by default is set to none.

$n\_components: int, \; default = None$
It is responsible for the count of components and by default is set to none which is the minimum of the number of all classes-1 and the number of features. Otherwise it can be set manually having in mind that the value must be integer and it must be equal or less to the same requirement as when it is set to none.

$store\_covariance: bool, \; default = False$
This parameter is a boolean and the default is False, but when it is True and the solver parameter is chosen to be ‘svd’ then it will explicitly calculate the within-class weighted covariance matrix. For the other methods there is no worries because it will calculate and store it automatically.

$tol: float, \; default = 1.0\,e^{-4}$
It is used only when the solver parameter is ‘svd’ and represents the absolute value used as a threshold for significance.

$covariance\_estimator: covariance \; estimator, \; default = None$ **[4]**
The default is none but If the covariance_estimator is not set to None, it is utilized to estimate the covariance matrices, rather than depending on the empirical covariance estimator, which may involve shrinkage. The object should possess a fit method and a covariance_ attribute, similar to the estimators found in sklearn.covariance. If it is None, the estimate is driven by the shrinkage parameter.

● **Illustration:**

First Run of Exploratory Data Analysis:

Plotting the histograms of the features in the dataset

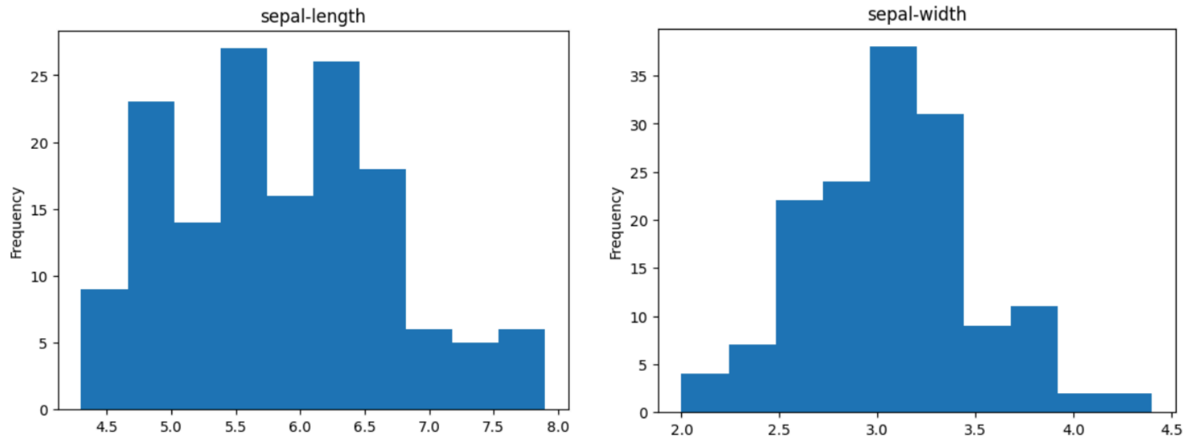


Fig.1. Histogram of Sepal Length and Width accordingly

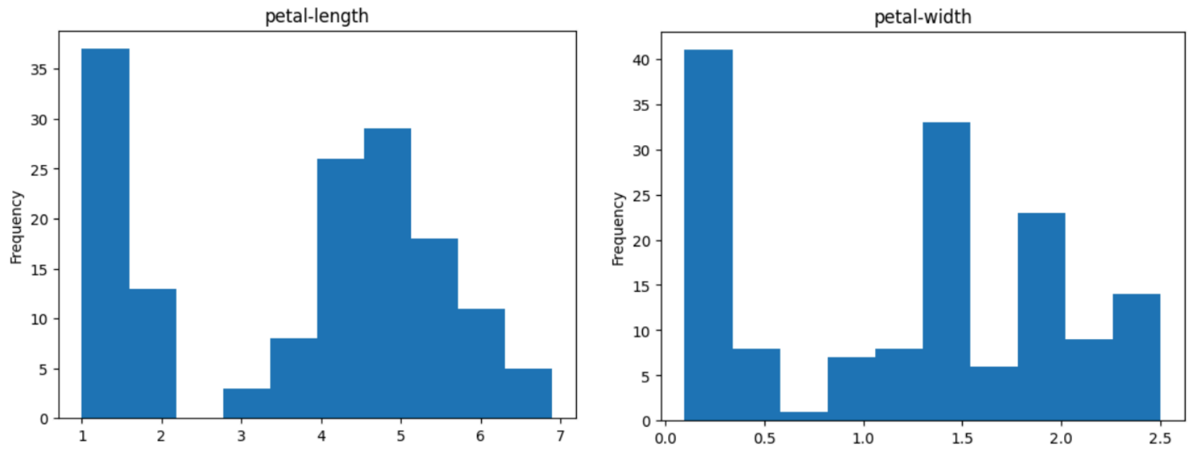


Fig.2. Histogram of Petal Length and Width accordingly

```
Accuracy : 0.8666666666666667
[[11  0  0]
 [ 4  9  0]
 [ 0  0  6]]
```

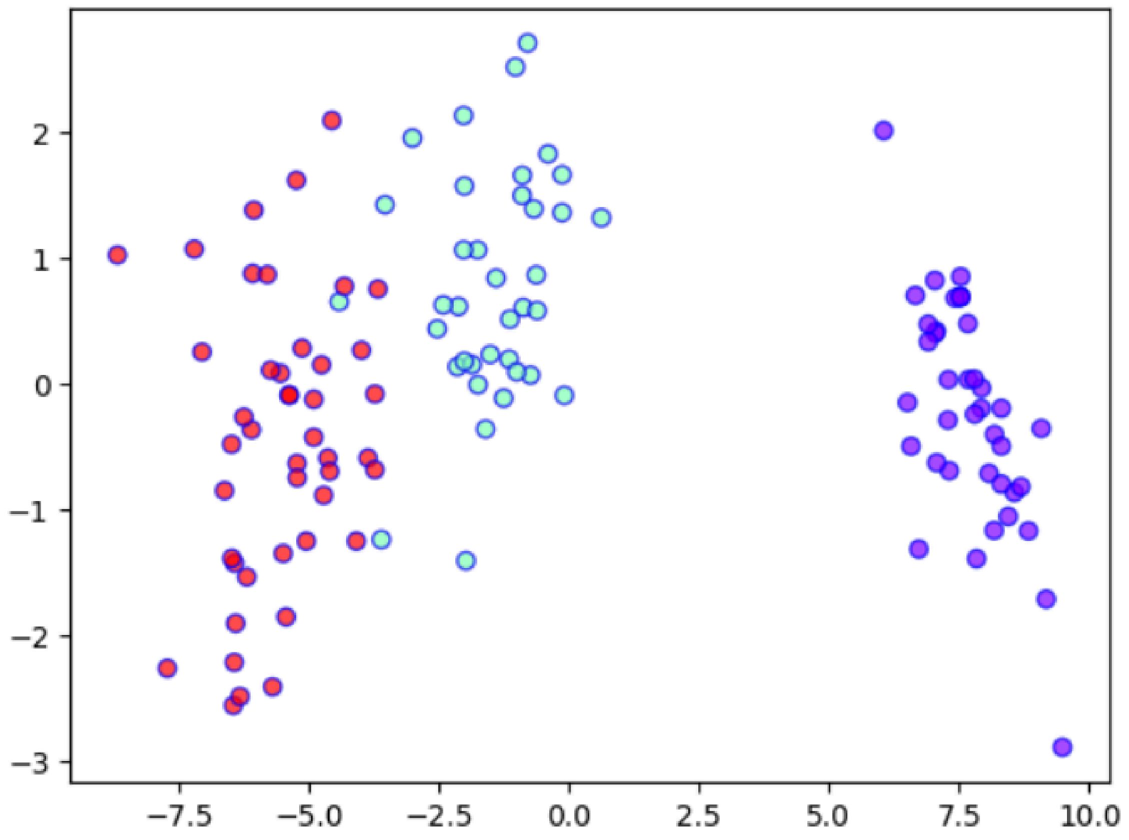


Fig.3. Classification after LDA applied to the dataset; Accuracy = 0.87 (NO lognormal transformation)

This is the result achieved from applying LDA to the dataset without log normalization. As we saw in the initial exploratory data analysis the data does NOT have a normal distribution and this is very important because by default the LDA assumes that the data has such distribution. This is why we tried to do some additional data preparation and apply lognorm transformation just to see if this is going to affect the final accuracy of the classification.

Second Run of Exploratory Data Analysis:

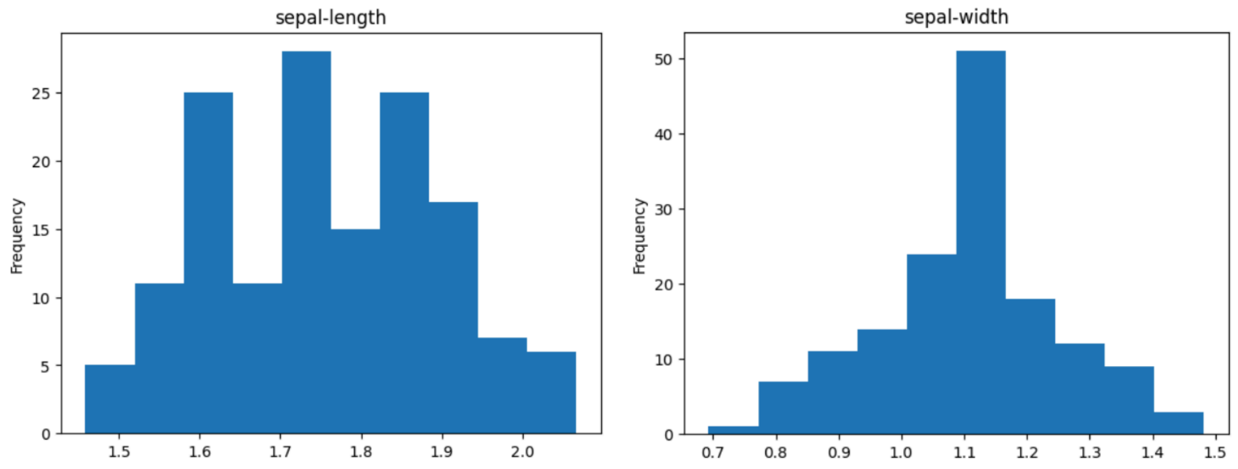


Fig.4. Histogram of Sepal Length and Width accordingly after lognorm transformation

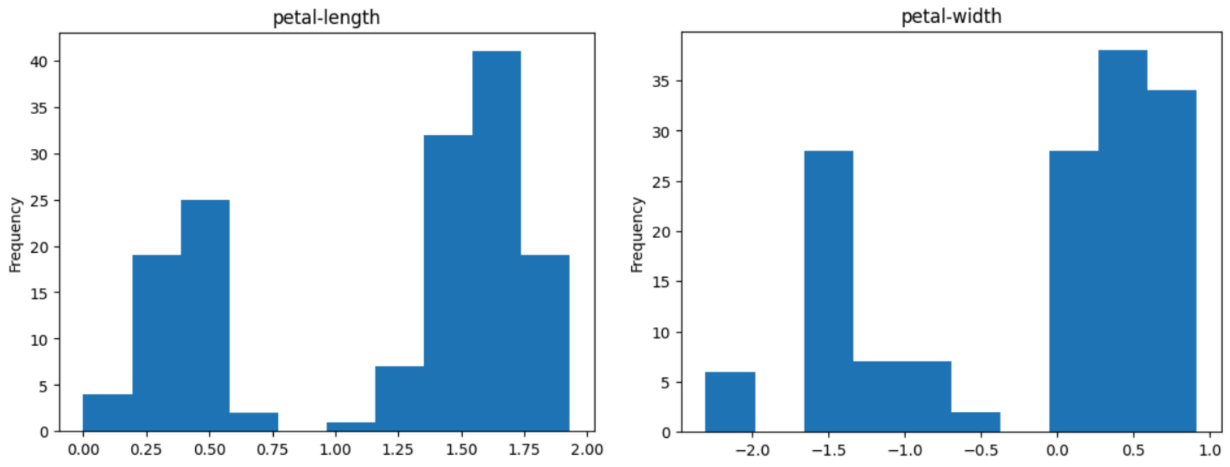


Fig.5. Histogram of Petal Length and Width accordingly after lognorm transformation

Still does NOT look like a normal distribution. This is maybe because the dataset is not big enough for the Central Limit Theorem to be in full power. Overall a log normal transformation makes the data to be closer to a Normal Distribution, which is important for LDA.

```
Accuracy : 1.0
[[11  0  0]
 [ 0 13  0]
 [ 0  0  6]]
```

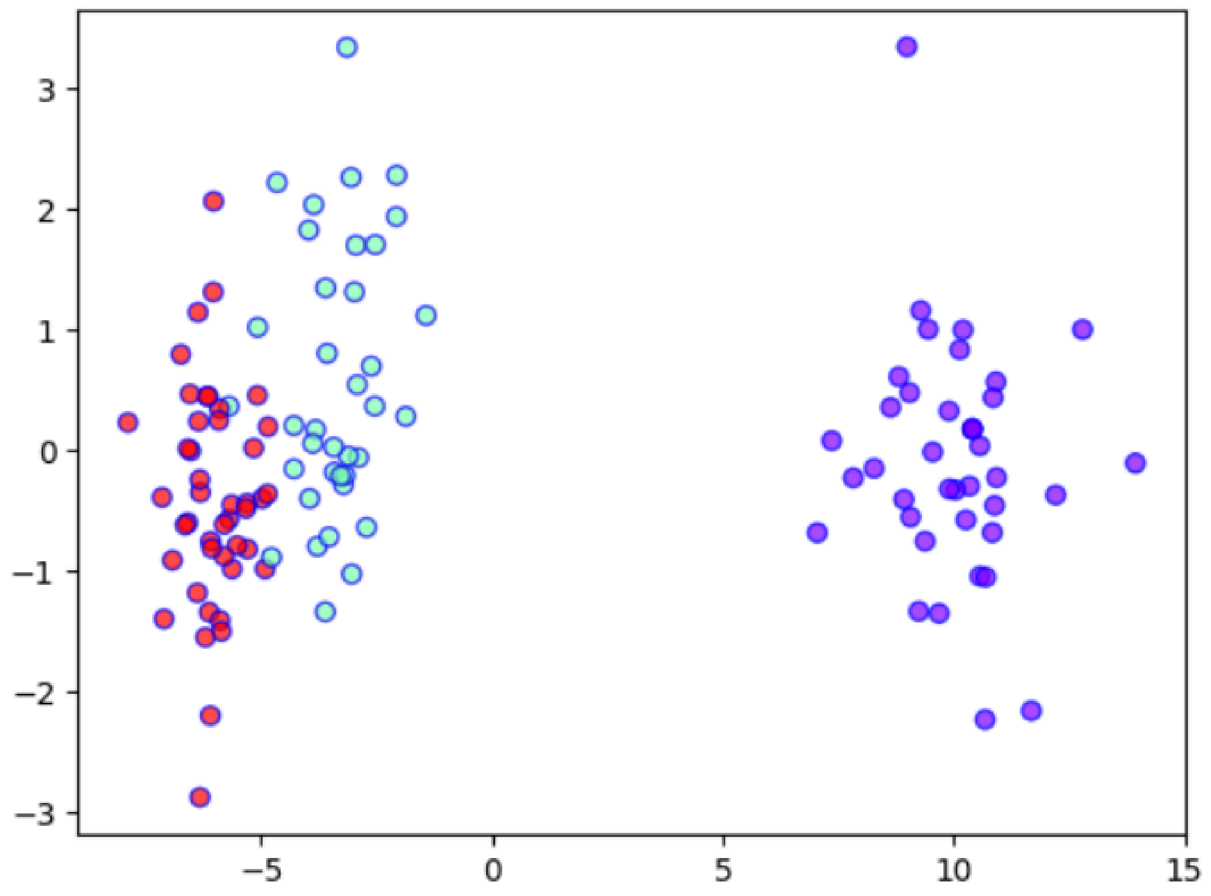


Fig.6. Classification after LDA applied to the dataset; Accuracy = 1 after lognormal transformation

Now we have a perfect classification with an accuracy = 1 and this is how we prove that LDA needs a Normally Distributed data (this is why we did the lognormal transformation) to do a better job.

● **Journal:**
Edward I. Altman and Robert A. Eisenbeis, Financial Applications of Discriminant Analysis: A Clarification, the Journal of Financial and Quantitative Analysis

● **Keywords: Create "tags" that identify this mode**
Machine Learning, Supervised Learning, LDA, Linear Discriminant Analysis, Classification, Dimensionality Reduction, Feature Extraction, Pattern Recognition, Data Analysis, Statistical Learning, Predictive Modeling, Data Mining, Multiclass Classification, Linear Algebra, Linear Transformation, Statistical Modeling

Support Vector Machines

Advantages:

- Effective in high-dimensional spaces.
- Memory-efficient as it uses a subset of training points (support vectors) for decision function.
- Versatile as it supports different kernel functions for non-linear decision boundaries.
- Resistant to overfitting, especially in high-dimensional spaces.

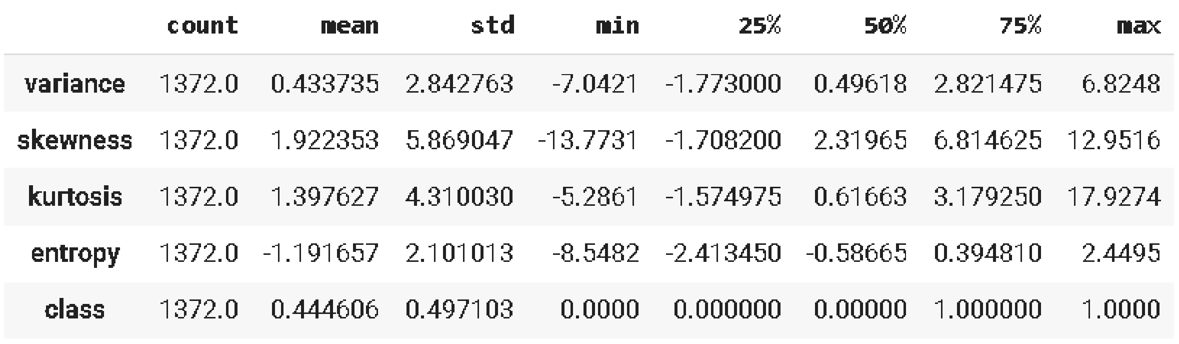
Basics:

Support Vector Machine is a supervised machine learning algorithm used for classification and regression tasks. It aims to find a hyperplane in an N-dimensional space (N being the number of features) that distinctly classifies data points into different classes.

Computation/Illustration/Visualization:

The study under the SVM model is to decide if the bank note is authentic or not. We import the bank note data from UCI database, explore if it's balanced data, the statistics of the input data including variance, skewness, kurtosis, entropy and class (binary 0 and 1), split 20/80 to test/training data, find the optimal parameters via Grid Search.

Descriptive statistics of the input data



	count	mean	std	min	25%	50%	75%	max
variance	1372.0	0.433735	2.842763	-7.0421	-1.773000	0.49618	2.821475	6.8248
skewness	1372.0	1.922353	5.869047	-13.7731	-1.708200	2.31965	6.814625	12.9516
kurtosis	1372.0	1.397627	4.310030	-5.2861	-1.574975	0.61663	3.179250	17.9274
entropy	1372.0	-1.191657	2.101013	-8.5482	-2.413450	-0.58665	0.394810	2.4495
class	1372.0	0.444606	0.497103	0.0000	0.000000	0.00000	1.000000	1.0000

From the histogram below, we can observe that variance has the normal distribution, compared with skewness, kurtosis, entropy. From the pairplot below, we may need non-linear SVM to make classification, instead of linear SVM.

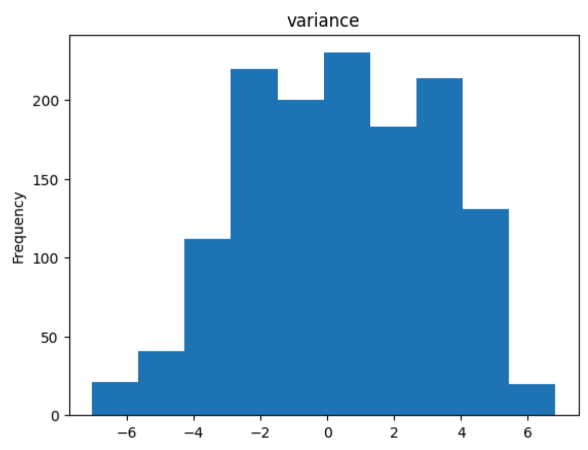


Fig.7. Histogram of variance

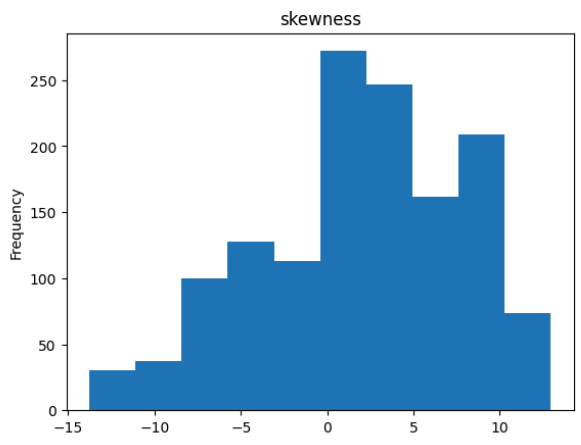


Fig.8. Histogram of Skewness

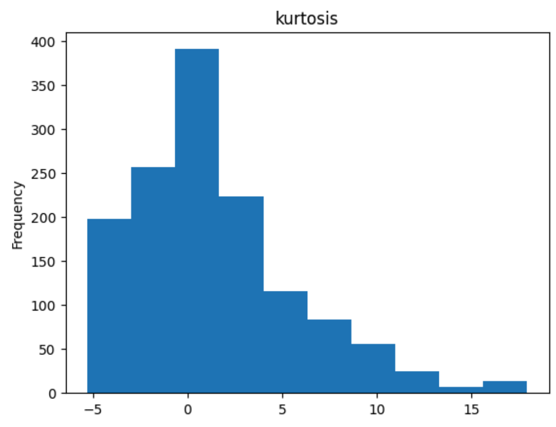


Fig.9. Histogram of Kurtosis

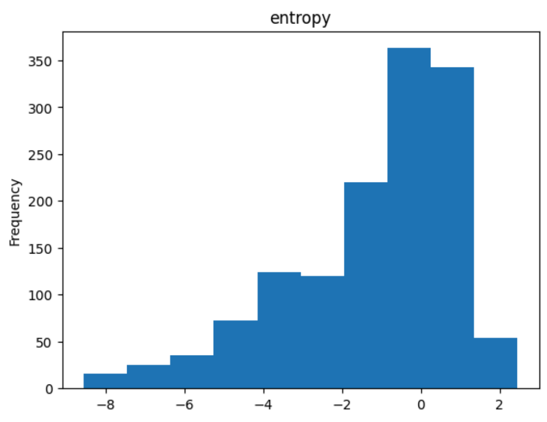


Fig.10. Histogram of Entropy

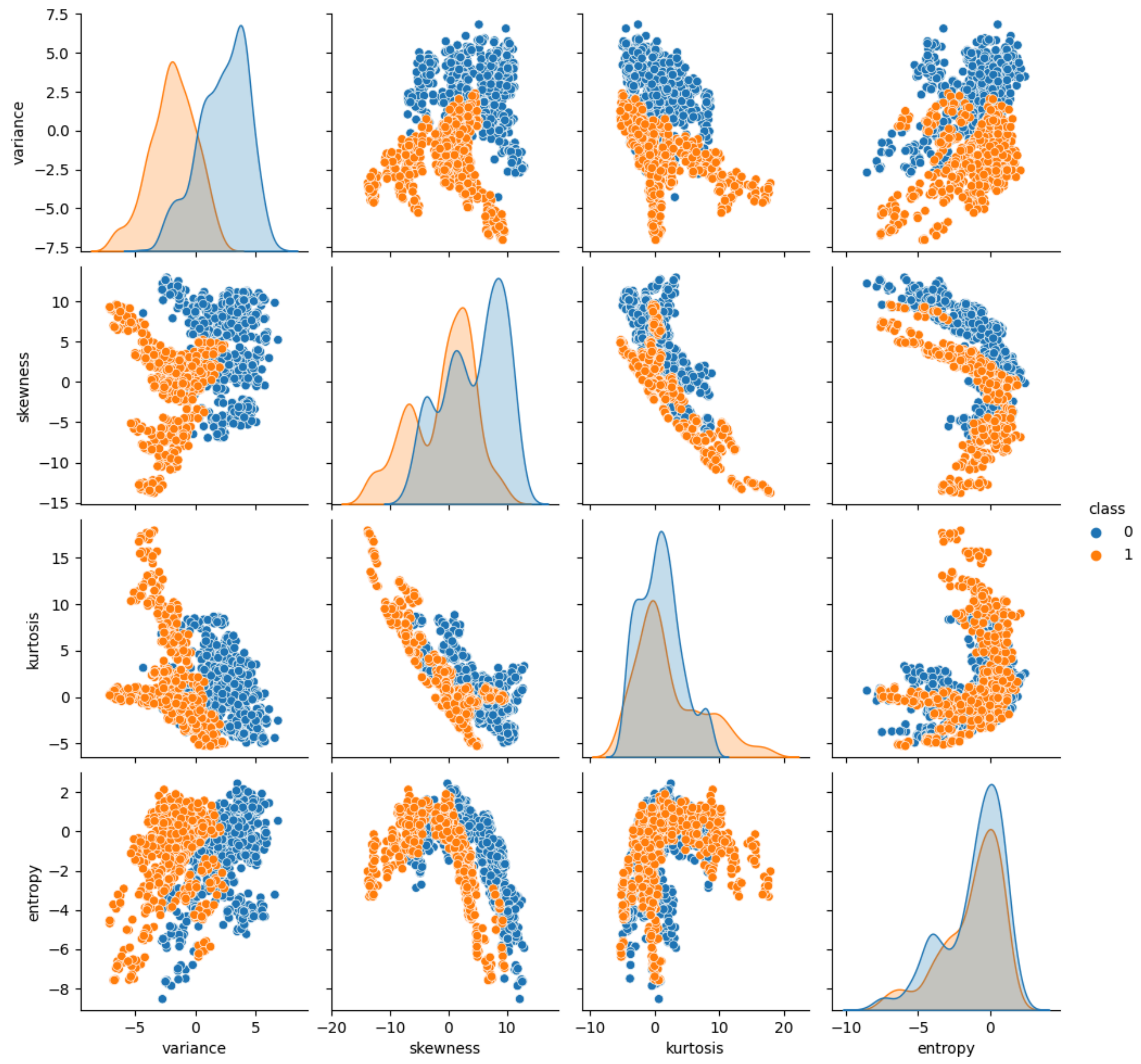


Fig.11. Pairplots of variance, skewness, kurtosis and entropy

From looking at the plots above, it's likely to have a non-linear kernel, instead of a linear one. To confirm our hypothesis, we apply grid search and found the following results:

```
The best model was: SVC(C=0.0001, gamma=10, kernel='poly')
The best parameter values were: {'C': 0.0001, 'gamma': 10, 'kernel': 'poly'}
The best f1-score was: 0.9989528795811518
```

To conclude, according to the results from grid search and the confusion matrix of different kernels, the best model doesn't have a linear kernel, but a nonlinear one, polynomial. We also plot the confusion matrix for linear, RBF and polynomial kernels.

Linear Kernel
svc = SVC(C=0.0001, gamma=10, kernel='linear')
Accuracy: 0.87

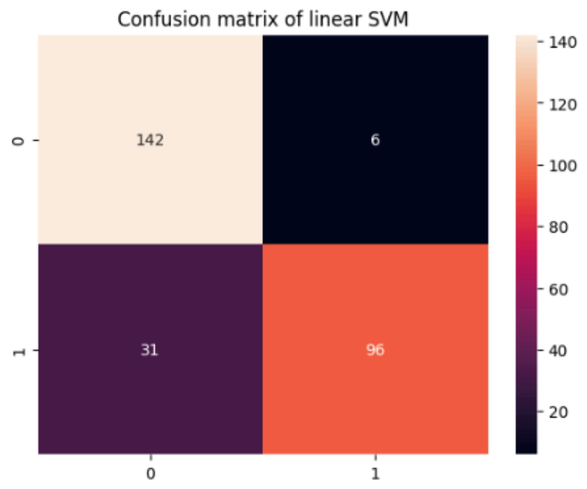


Fig.12. Confusion matrix of Linear SVM: accuracy at 0.87

RBF Kernel
svc = SVC(C=1, gamma=1, kernel='rbf')
Accuracy: 0.9979166666666666

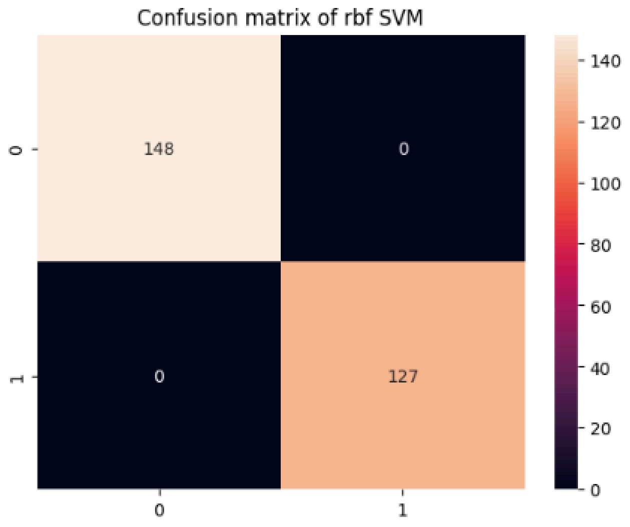


Fig.13. Confusion matrix of RBF SVM: accuracy at 0.9979

```
Polynomial Kernel
svc = SVC(C=0.0001, gamma=10, kernel='poly')
Accuracy:  0.9989528795811518
```

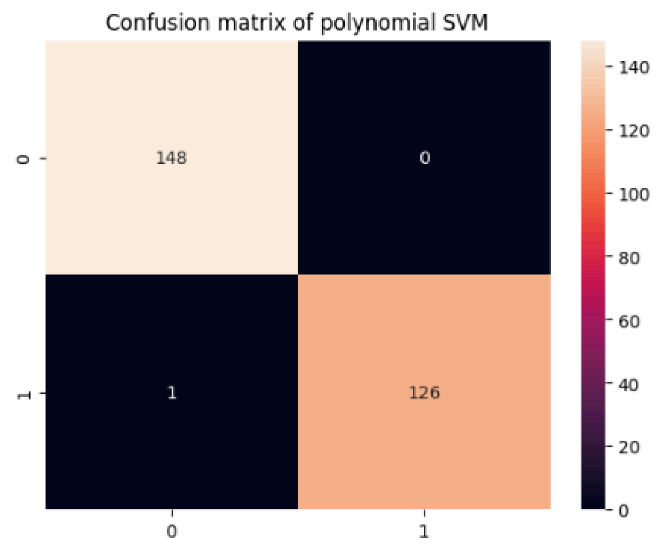


Fig.14. Confusion matrix of Polynomial SVM: accuracy at 0.9990

Accuracy under train data:

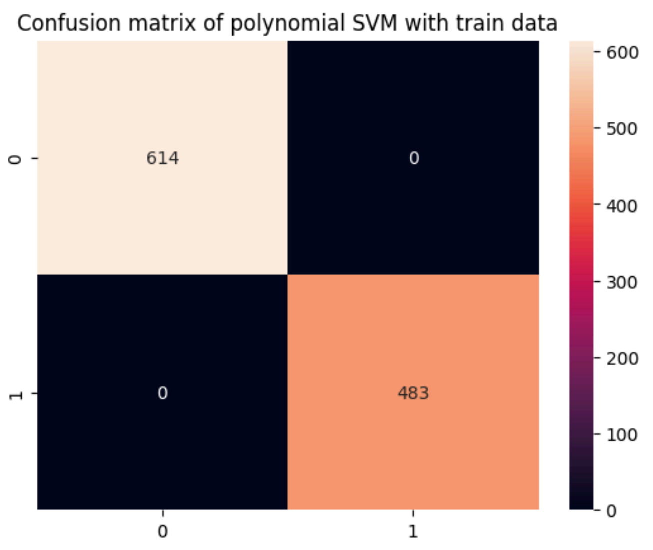


Fig.15. Confusion matrix of Polynomial SVM: accuracy at 1.00

Disadvantages:

- Not suitable for larger datasets as the training time can be high.
- Sensitivity to noisy data and outliers.
- Choosing an appropriate kernel function and tuning hyperparameters can be challenging.

Equations:

Hyperplane equation (the equation of a straight line in p-dimensions):

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = 0$$

Where:

β_0 *is the intercept*

β_1 *to* β_p *are the coefficients*

Or simplified:

$$\sum_{j=1}^{p} \beta_j X_j + \beta_0 = 0$$

The decision function for a linear SVM is given by:

$$y_i\left(\sum_{j=1}^{p} \beta_j X_j + \beta_0\right) > D$$

Where:

y_i *is the class label for the* i^{th} *observation*

D *is the perpendicular distance of the hyperplane to the support vector*

Kernels:

If the data is not linearly separable, then we can use a kernel, which is a method that represents a data transformation into higher dimension so that we could do a better split of data.
The hyperplane solution depends on the inner product of the observations. Then the case for a 2 dimentional kernel transformation looks like that:

$$Z = \sum_{j=1}^{2} \langle X_{j_i}, X_{j_{i'}} \rangle = K(X_i, X_{i'})$$

A linear support vector formula:

$$y(x) = \beta_0 + \sum_{i}^{n} \alpha_i \langle X, X_i \rangle$$

The Gaussian RBF kernel's formula:

$$K(X_i, X_{i'}) = exp(-\gamma ||X_i - X_{i'}||^2)$$

Where:

$||X_i - X_{i'}||^2$ *is the Euclidean distance*

γ *is a hyperparameter that is given by the user and if it is too large it can lead to overfitting*

Features:

- Works well in high-dimensional spaces.
- Effective in cases where the number of dimensions is greater than the number of samples.
- Handles non-linear decision boundaries through kernel trick.

Guide:

Inputs: Training data (X), corresponding labels (y), kernel type, and hyperparameters.
Outputs: A decision function that classifies new data points.

Hyperparameters:

- C (regularization parameter): the smaller the value of C, the larger the margin. L2 regularization or ridge regularization
- Kernel type (linear, polynomial, radial basis function (RBF), etc.)
- Gamma (kernel coefficient for RBF): The higher the value of gamma, the closer are the points that are considered for the decision boundary, and the lower the gamma, the farther points are also considered for choosing the decision boundary.

Journal Reference:

Kumar, Manish and Thenmozhi, M., Forecasting Stock Index Movement: A Comparison of Support Vector Machines and Random Forest. Indian Institute of Capital Markets 9th Capital Markets Conference Paper, Available at SSRN: https://ssrn.com/abstract=876544 or http://dx.doi.org/10.2139/ssrn.876544

Huang, Allen H. and Wang, Hui and Yang, Yi, FinBERT - A Large Language Model for Extracting Information from Financial Text (July 28, 2020). Contemporary Accounting Research, Forthcoming, Available at SSRN: https://ssrn.com/abstract=3910214 or http://dx.doi.org/10.2139/ssrn.3910214

Singh, Siddharth and Kaushik, Mayank and Gupta, Ambuj and Malviya, Anil Kumar, Weather Forecasting Using Machine Learning Techniques (March 11, 2019). Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE) 2019, Available at SSRN: https://ssrn.com/abstract=3350281 or http://dx.doi.org/10.2139/ssrn.3350281

Georgoula, Ifigeneia and Pournarakis, Demitrios and Bilanakos, Christos and Sotiropoulos, Dionisios and Sotiropoulos, Dionisios and Giaglis, George M., Using Time-Series and Sentiment Analysis to Detect the Determinants of Bitcoin Prices (May 17, 2015). Available at SSRN: https://ssrn.com/abstract=2607167 or http://dx.doi.org/10.2139/ssrn.2607167

Paul, Sanmoy and Acharya, Sameer Kumar, A Comparative Study on Facial Recognition Algorithms (December 21, 2020). e-journal - First Pan IIT International Management Conference – 2018, Available at SSRN: https://ssrn.com/abstract=3753064 or http://dx.doi.org/10.2139/ssrn.3753064

Huerta, Ramon and Elkan, Charles and Corbacho, Fernando, Nonlinear Support Vector Machines Can Systematically Identify Stocks with High and Low Future Returns (September 6, 2012). Algorithmic Finance (2013), 2:1, 45-58, Available at SSRN: https://ssrn.com/abstract=1930709 or http://dx.doi.org/10.2139/ssrn.1930709

Palanivel, Kodimalar and Surianarayanan, Chellammal, An Approach for Prediction of Crop Yield Using Machine Learning and Big Data Techniques (2019). International Journal of Computer Engineering and Technology 10(3), pp. 110-118, 2019, Available at SSRN: https://ssrn.com/abstract=3555087

TIWARI, MONIKA and Bharuka, Rashi and Shah, Praditi and Lokare, Reena, Breast Cancer Prediction Using Deep Learning and Machine Learning Techniques (March 22, 2020). Available at SSRN: https://ssrn.com/abstract=3558786 or http://dx.doi.org/10.2139/ssrn.3558786

Chen, James Ming, An Introduction to Machine Learning for Panel Data (October 23, 2020). International Advances in Economic Research, Vol. 27, 2021, Available at SSRN: https://ssrn.com/abstract=3717879 or http://dx.doi.org/10.2139/ssrn.3717879

Chakraborty, Chiranjit and Joseph, Andreas, Machine Learning at Central Banks (September 1, 2017). Bank of England Working Paper No. 674, Available at SSRN: https://ssrn.com/abstract=3031796 or http://dx.doi.org/10.2139/ssrn.3031796

Keywords:
Stock Market, Ensemble Methods, Genetic Algorithm, Ghana-Stock-Exchange, Random Forest, Decision Trees, Support Vector Machine, Neural Networks, Stock Market Prediction, Trading Strategies; Support vector machine, Random forest, Forecasting, Stock index; Deep Learning; Large Language Model; Transfer Learning; Interpretable Machine Learning; Sentiment Classification; Environment, Social, and Governance (ESG); Machine Learning, Weather Forecasting, Artificial Neural Network, Time Series Analysis using Recurrent Neural Network, Time Series Analysis; Bitcoins, error correction, machine learning, sentiment analysis; Eigenvalues; Haar Cascades Facial Recognition Principal Component Analysis, Convolutional Neural Network (CNN), K-Nearest Neighbour (KNN); reinforcement learning, sector neutral; STA, IISTA, Image Restoration, Inverse Problems, l0 Norm, l1 Norm, l2 Data Fidelity Term, Regularization Function, Total Variation; bias-variance tradeoff, decision trees, random forests, extra trees, XGBoost, learning ensembles, boosting, support vector machines, neural networks; artificial intelligence, big data, econometrics, forecasting, inflation, financial markets, banking supervision, financial technology

Neural Networks

● **Advantages:**

- It's a very good model for datasets that have complex patterns.
- It can be easily adapted to new examples.
- Parallel processing in Neural Networks can be done.
- It requires less formal statistical training.

● **Basics:**

Neural Network is a model in Machine Learning which is mostly inspired by replicating the human brain with a computerized algorithm. It's basically a model to establish the relationship between input and output the way the human brain establishes. Its process is simple to understand but difficult to implement which involves input, analyzing or calculation/processing of inputs and then making the decision or giving the output. [5],[6],[8]

● **Computation:**

Using the neural network algorithm to predict the monthly returns of DJI futures using economic indicators from FRED as predictors and formatted the data for predictive analytics scenarios. [7]

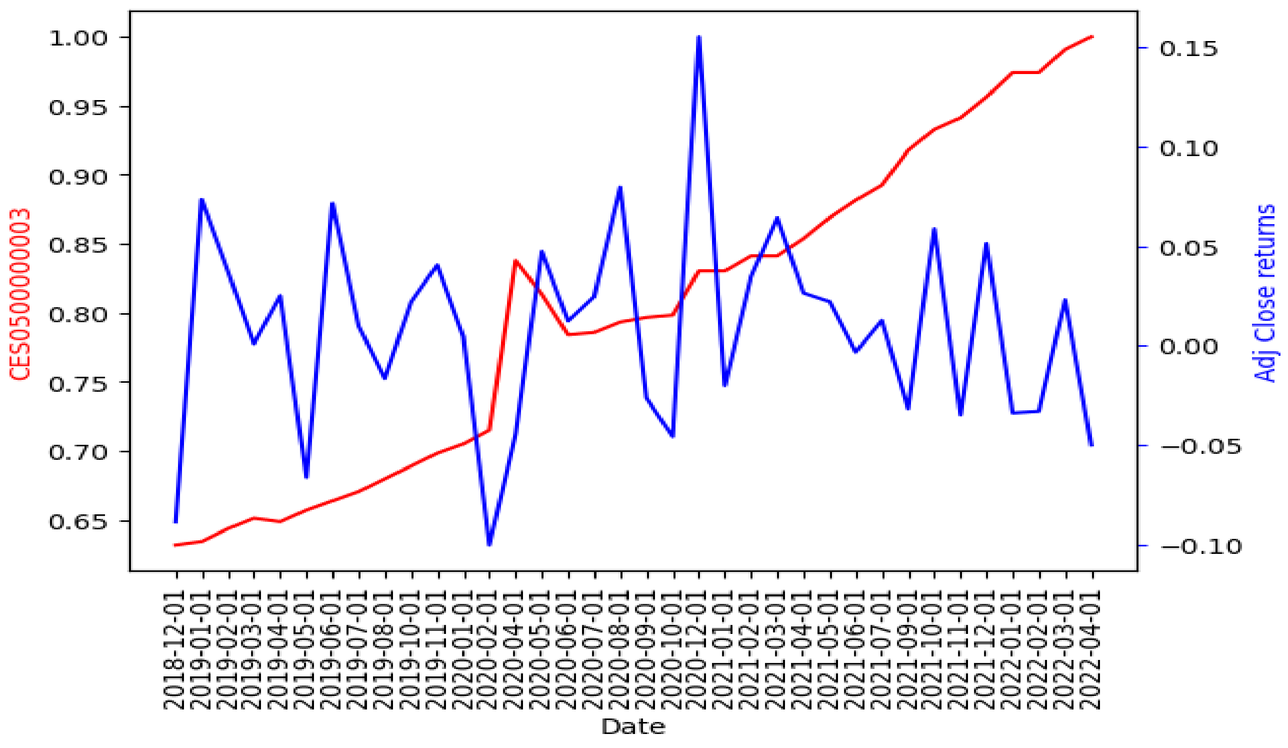


Fig.16. Relation between predictor average hourly earnings of all employees vs DJI futures return-

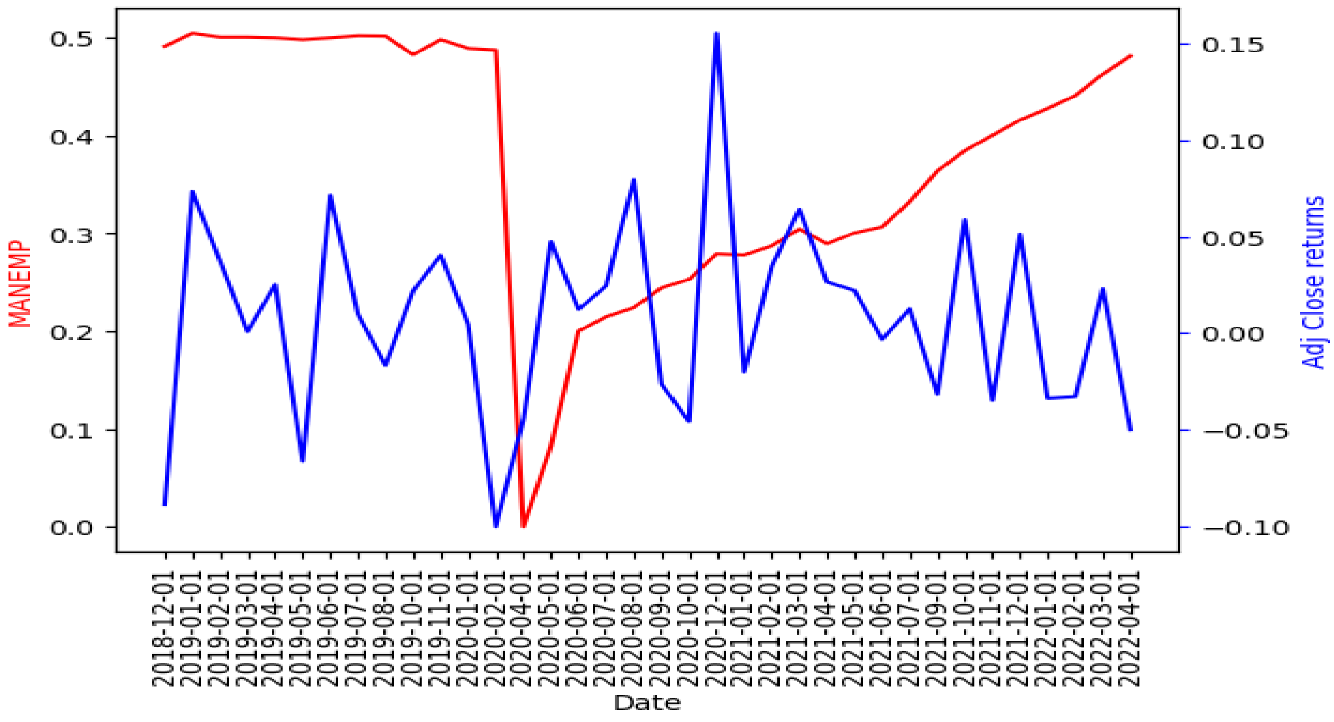


Fig.17. Relation between all employees manufacturing sector vs DJI futures return

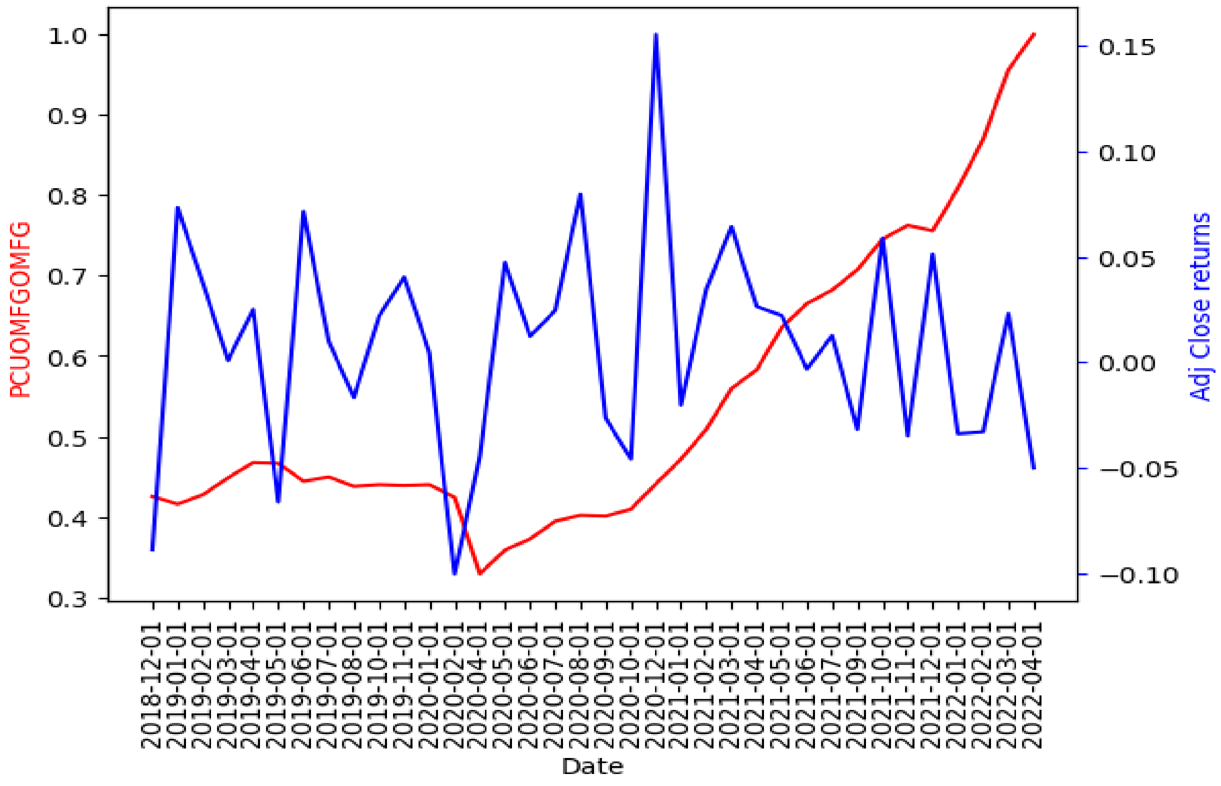


Fig.18. Relation between predictor producer price index by industry vs DJI futures return

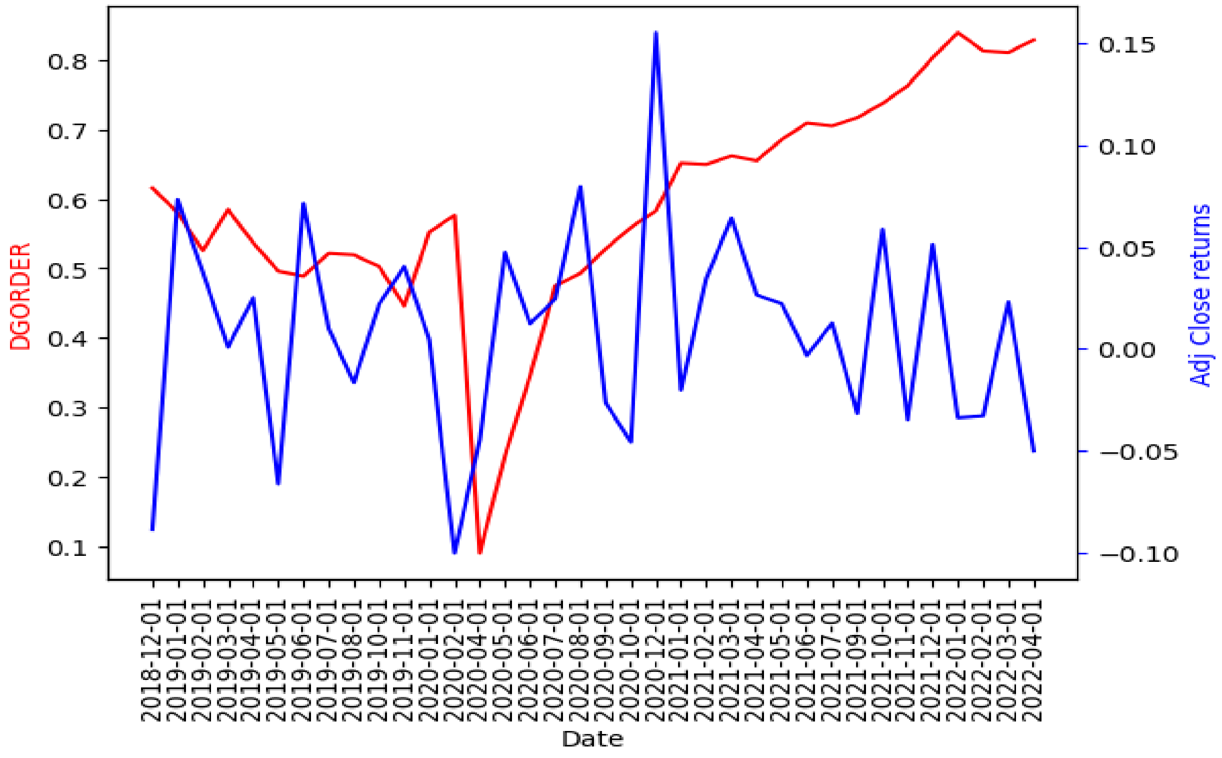


Fig.19. Relation between predictor durable goods new orders vs DJI futures return

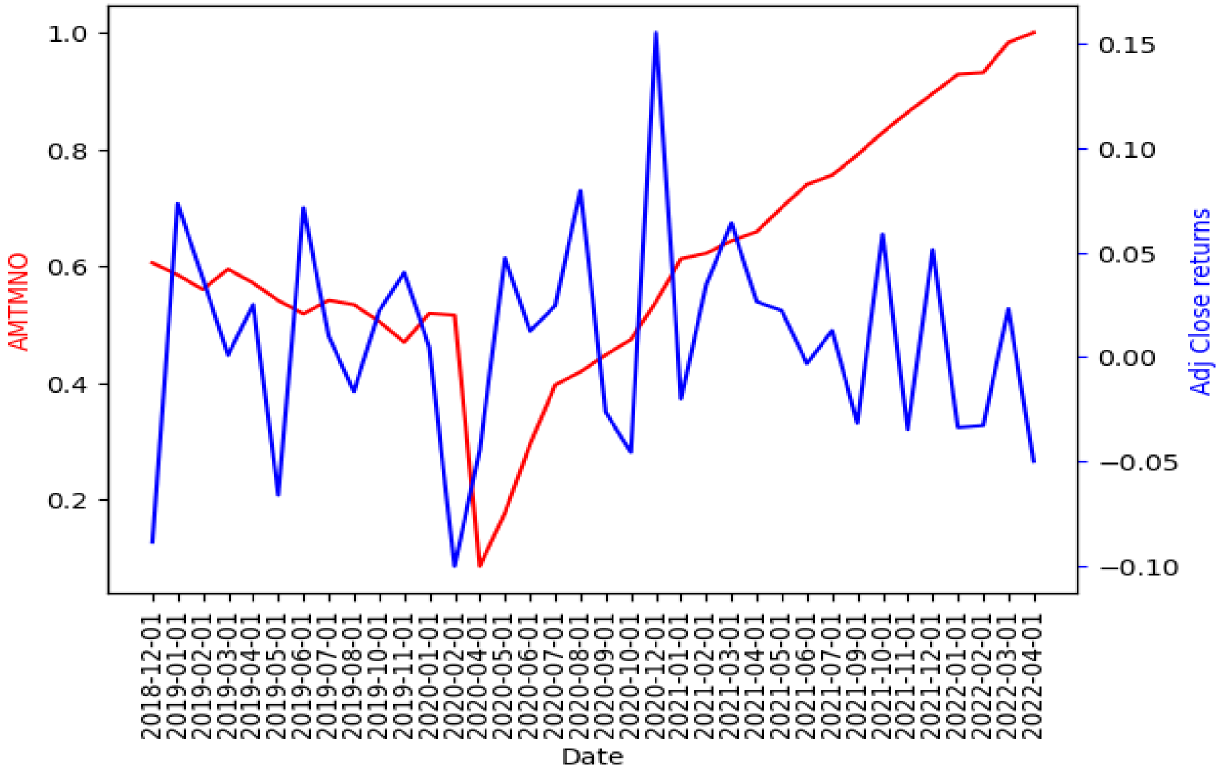


Fig.20. Relation between predictor total manufacturing vs DJI futures return

● **Disadvantages:**

- It's computationally a very vast model.
- It works well for large data sets and, it's time consuming.
- It has more issues of overfitting.
- Understanding and making interpretation of Neural Network is challenging.

● **Equations:**

Equations that summarize how the model works-

Input Layer- X_0

Hidden Layer- X_1

Output Layer- Y

In forward propagation Neural Networks we have the following calculation for Hidden Layer-

X_1= f($X_0.W_0 + b_0$)

Where Xo.Wo is a matrix multiplication/dot product of Input layer and Weights. bo is a bias term and f is an activation function.

Then, for calculation of output layers we have the following equation-

Y= f($X_1.W_1 + b_1$)

Or,

Y= f(f($X_0.W_0 + b_0$).$W_1 + b_1$)

Finally, predicted output Y is compared to the actual target T using a loss function L as:

L (Y, T)

Where:

L – Loss function

Y – predicted output

T – target value

There are many activation functions are used in Neural networks and among them few are as below:

- Sigmoid function:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

- Tangent (Tanh) function:

$$Tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

- Rectified Linear unit (ReLU) function:

$$Relu(x) = max(0, x)$$

● **Features:**

- It works well with the complex data set
- It can deal with noise and outliers with the help of non-linear activation functions like ReLU
- It has the ability of data augmentation which helps to create new training examples by applying transformation to the existing data and it helps to reduce noise.
- It works well with real life situations where data are having uncertainties and variations.

● **Guide:**

Inputs- the input layer of Neural network are neurons of the attributes of input data.

Output- the predictions or result generated by the model based on the processing at hidden layers.

Hidden- this is the layer between input and output of Neural Network in which the learning and extracting of input data are performed with the help of weight matrix.

● **Hyperparameters:**

Learning rate- the amount of change to the model relative to the change in error.

Activation function- a function that is applied to the output of a neuron in neural network

Epochs- the number of times the entire training dataset pass through the learning algorithm

Size of the layer- the number of layers in input, hidden and output nodes

Regularization- it's used to reduce impact of overfitting

● **Illustration:**

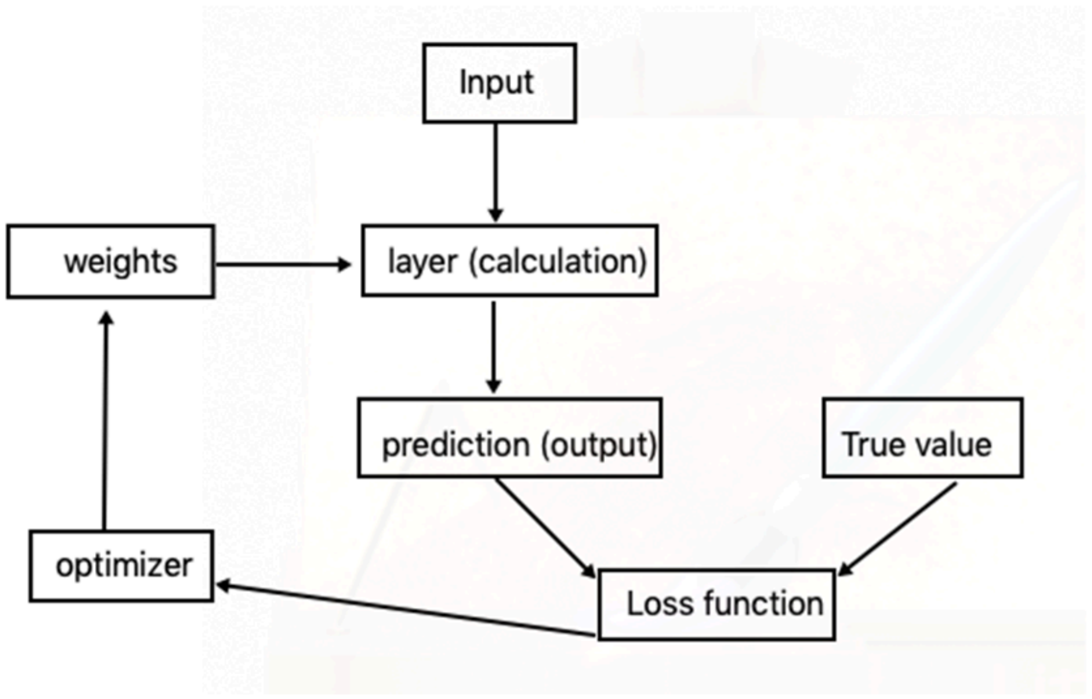


Fig.21. Neural Networks flow diagram

● **Journal:**

1. https://www.ibm.com/topics/neural-networks

2. Introduction to Machine Learning, Neural Networks and Deep Learning -An article by Rene Y. Choi; Aaron S. Coyner; Jayashree Kalpathy-Cramer; Michael F. Chiang; J. Peter Campbell

3. Activation Functions in Neural Networks- an article by Sagar Sharma

https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

● **Keywords:**

Machine Learning, Neural Networks, Hidden Layers, Optimizer, Weights, Loss function, Activation function, Forward propagation, Backward Propagation, Bias, Gradient descent, Vanishing Gradient, Sigmoid, Tangent, optimization, overfitting, noise, Learning rate, Epochs, Batch size, Predictors, Attributes, Pattern, Momentum, Hyperparameter, Algorithm

Technical Section

LDA

The LDA's hyperparameters are tuned in our case by the GridSearch method. We chose to tune the following LDA's hyperparameters: solver and shrinkage Where for the solver method the options are: 'svd', 'lsqr', 'eigen' and they are detailed explained in the previous section. In our case the optimization showed that the best setting is the default one: 'svd'. This is totally normal and this is why this is the default setting, because in most of the cases this would be the best choice.
For the other hyperparameter: shrinkage, the best choice out of the possible settings: 'auto', 'float', or 'default', where our possible values were: None, 'auto', '0.1', '0.5', '0.9', the choice was None, which again is the default one.

Support Vector Machine

As we have learned from Step 1-3, SVM has several hyperparameters such as regularization parameter C, kernel and gamma for RBF kernel. We find the optimal hyperparameters via grid search along with cross-validation, including selecting the list of hyperparameters grid, splitting 20/80 to test and training datasets, performing grid search and evaluating on test set.

First, we specify a range of values for each hyperparameter. For example, for the regularization parameter C, we define a set of values like [0.0001, 1, 10]. For the gamma parameter in an RBF kernel, we consider values like [1, 10, 100]. We define a set of kernels such as radial basis function, sigmoid, linear and polynomial.

Second, with splitting the training data into 5 subsets (folds), training the model on 4 folds, and validating on the remaining fold. Repeat this process 5 times, rotating the validation fold each time, we can use the average performance across all folds to assess the model's performance.

We perform Grid Search and evaluate test sets. As a result, we obtain the following optimal hyperparameter values with the highest F1-score at 0.9989, which has the regularization parameter C at 0.0001, gamma at 10, and polynomial kernel:

```
The best model was: SVC(C=0.0001, gamma=10, kernel='poly')
The best parameter values were: {'C': 0.0001, 'gamma': 10, 'kernel': 'poly'}
The best f1-score was: 0.9989528795811518
```

In addition, we also compared the accuracy score with train data. There is no overfitting issues observed.

Neural Networks

Tuning in Neural Network Hyperparameter is done with finding of optimal learning rate which is in between of 0.01 to 1 so that issue of non-convergence can be reduced. The other tuning we did in the Epochs to balance the number of times the algorithm work and the time consumption. Here in the example, we tuned the Epochs from 10 to 100. Also, we tuned the momentum as 0.9 to balance the non-convergence and number of epochs. Also, another tuning is done on batch size which is of size 8 which means the common sizes are in the increment of 8.

Marketing Alpha

LDA

Benefits:

It offers superior interpretability compared to PCA.
Its simplicity and computational efficiency make it a preferred algorithm.
It works well even with multicollinearity.
It is even capable of handling large datasets, including those with more features than training samples.

We showed that when LDA is applied with all it assumptions honored it achieves very good results. The comparison is done with 2 LDA runs: one without lognorm transformation to the dataset and the second after the transformation which is one of the main LDA's assumptions (the data to be normally distributed). As we know lognorm transformation makes the data distribution to be one idea closer to a normal distribution and the results are as follows: fig.3 Classification after LDA applied to the dataset; Accuracy = 0.87 (NO lognormal transformation) and fig. Classification after LDA applied to the dataset; Accuracy = 1 after lognormal transformation

Support Vector Machine:

Advantages:

- Effective in high-dimensional spaces.
- Memory-efficient as it uses a subset of training points (support vectors) for decision function.
- Versatile as it supports different kernel functions (eg. sigmoid, radial basis function and polynomial etc) for non-linear decision boundaries (from the pairplot below, we make the hypothesis that we may need a non-linear kernel).
- Resistant to overfitting, especially in high-dimensional spaces.

Features:

- Works well in high-dimensional spaces.
- Effective in cases where the number of dimensions is greater than the number of samples.
- Handles non-linear decision boundaries through kernel trick.

From our previous example, we find that SVM can select linear, sigmoid, RBF and polynomial kernels to handle both linear and non linear decision boundaries. From the confusion matrix below with high accuracy scores with train and test sets, we can observe that SVM can be resistant to overfitting. In general, SVM can handle data effectively in high dimensionality situations, and be robust to outliers. See **Fig.11, Fig.14.,**
and **Fig.15.** for refference.

Neural Networks:

The Neural Network model works well for datasets which are having a complex pattern. It can adapt and learn from the input which allows to improve the performance by optimizing the processing in hidden layers. Since it has the ability to process multiple inputs simultaneously it's a good fit where parallel processing is required. Also, it has a better ability to handle noise hence it's more useful in real world applications where input data has more uncertainty and variability. Neural networks is pattern centric model so it's good for patter recognition applications like image recognition, speech recognition etc.

Learn more

LDA:

Joy, O. Maurice, and John O. Tollefson. “On the Financial Applications of Discriminant Analysis.” *Journal of Financial and Quantitative Analysis*, vol. 10, no. 5, 1975, pp. 723-7391.

Edward I. Altman, Robert A. Eisenbeis; Financial Applications of Discriminant Analysis: A Clarification; Published online by Cambridge University Press in the Journal of Financial and Quantitative Analysis: 06 April 2009

Support Vector Machine:

Kumar, Manish and Thenmozhi, M., Forecasting Stock Index Movement: A Comparison of Support Vector Machines and Random Forest. Indian Institute of Capital Markets 9th Capital Markets Conference Paper, Available at SSRN: https://ssrn.com/abstract=876544 or http://dx.doi.org/10.2139/ssrn.876544

Huang, Allen H. and Wang, Hui and Yang, Yi, FinBERT - A Large Language Model for Extracting Information from Financial Text (July 28, 2020). Contemporary Accounting Research, Forthcoming, Available at SSRN: https://ssrn.com/abstract=3910214 or http://dx.doi.org/10.2139/ssrn.3910214

Singh, Siddharth and Kaushik, Mayank and Gupta, Ambuj and Malviya, Anil Kumar, Weather Forecasting Using Machine Learning Techniques (March 11, 2019). Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE) 2019, Available at SSRN: https://ssrn.com/abstract=3350281 or http://dx.doi.org/10.2139/ssrn.3350281

Georgoula, Ifigeneia and Pournarakis, Demitrios and Bilanakos, Christos and Sotiropoulos, Dionisios and Sotiropoulos, Dionisios and Giaglis, George M., Using Time-Series and Sentiment Analysis to Detect the Determinants of Bitcoin Prices (May 17, 2015). Available at SSRN: https://ssrn.com/abstract=2607167 or http://dx.doi.org/10.2139/ssrn.2607167

Paul, Sanmoy and Acharya, Sameer Kumar, A Comparative Study on Facial Recognition Algorithms (December 21, 2020). e-journal - First Pan IIT International Management Conference – 2018, Available at SSRN: https://ssrn.com/abstract=3753064 or http://dx.doi.org/10.2139/ssrn.3753064

Huerta, Ramon and Elkan, Charles and Corbacho, Fernando, Nonlinear Support Vector Machines Can Systematically Identify Stocks with High and Low Future Returns (September 6, 2012). Algorithmic Finance (2013), 2:1, 45-58, Available at SSRN: https://ssrn.com/abstract=1930709 or http://dx.doi.org/10.2139/ssrn.1930709

Palanivel, Kodimalar and Surianarayanan, Chellammal, An Approach for Prediction of Crop Yield Using Machine Learning and Big Data Techniques (2019). International Journal of Computer Engineering and Technology 10(3), pp. 110-118, 2019, Available at SSRN: https://ssrn.com/abstract=3555087

TIWARI, MONIKA and Bharuka, Rashi and Shah, Praditi and Lokare, Reena, Breast Cancer Prediction Using Deep Learning and Machine Learning Techniques (March 22, 2020). Available at SSRN: https://ssrn.com/abstract=3558786 or http://dx.doi.org/10.2139/ssrn.3558786

Chen, James Ming, An Introduction to Machine Learning for Panel Data (October 23, 2020). International Advances in Economic Research, Vol. 27, 2021, Available at SSRN: https://ssrn.com/abstract=3717879 or http://dx.doi.org/10.2139/ssrn.3717879
Chakraborty, Chiranjit and Joseph, Andreas, Machine Learning at Central Banks (September 1, 2017). Bank of England Working Paper No. 674, Available at SSRN: https://ssrn.com/abstract=3031796 or http://dx.doi.org/10.2139/ssrn.3031796

Neural Network:

https://www.ibm.com/docs/en/wmla/1.2.3?topic=features-hyperparameter-tuning

Precision education with statistical learning and deep learning: a case study in Taiwan by Shuo-Chang Tsai, Cheng-Huan Chen, Yi-Tzone Shiao, Jin-Shuei Ciou & Trong-Neng Wu- https://link.springer.com/article/10.1186/s41239-020-00186-2

Artificial Neural Network by Sun-Chong Wang published in the part of The Springer International Series in Engineering and Computer Science book series (SECS, Volume 743)- https://link.springer.com/chapter/10.1007/978-1-4615-0377-4_5

State-of-the-art in artificial neural network applications: A survey: an article by Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Humaira Arshad- https://www.cell.com/heliyon/pdf/S2405-8440(18)33206-7.pdf

https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-a-neural-network/

Comparing Models

Features of Models	LDA	SVM	NN
Data Requirements	LDA has the assumption of normally distributed classes and equal covariance matrices.	SVMs work well especially when the number of features is not too high.	NN can handle diverse data types but may require a large amount of labeled data for training, and they can benefit from careful preprocessing.
Handling Non-Linearity	Due to its linearity assumption, LDA may not perform well when the decision boundary is highly non-linear.	SVM can handle both linear and non-linear decision boundaries via selecting appropriate kernels.	NN can handle nonlinear situations well
Dimensionality Reduction	LDA can be used for dimensionality reduction by projecting the data onto a lower-dimensional subspace while maximizing class separability.	SVM can't perform dimensionality reduction, but it can handle high-dimensional data effectively.	NN may implicitly perform dimensionality reduction through the learning process.
Interpretability	Most interpretable	Less interpretable	Least interpretable
Sensitivity to Outliers	LDA is sensitive to outliers because its method is to find a decision boundary that maximizes class separability.	SVMs are generally robust to outliers, especially when using a soft-margin approach.	NN can be sensitive to outliers, particularly during training.
Computational Complexity	It is computationally efficient due to the closed-form solution.	Although it can be computationally expensive, especially with large datasets, efficient solvers are available for SVM.	It can require powerful hardware and longer training times due to more complexity.

REFERENCES:

1. ML | Linear Discriminant Analysis; https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/
2. Linear Discriminant Analysis: Definition, Working, and Applications; https://intellipaat.com/blog/linear-discriminant-analysis/
3. Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems" (PDF). Annals of Eugenics. 7 (2): 179–188. doi:10.1111/j.1469-1809.1936.tb02137.x. hdl:2440/15227
4. scikit-learn 1.3.2 documentation; sklearn.discriminant_analysis.LinearDiscriminantAnalysis; https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html
5. Institute for Clinical Evaluative Sciences, North York, Ontario, Canada. https://pubmed.ncbi.nlm.nih.gov/8892489/#:~:text=Neural%20networks%20offer%20a%20number,the%20availability%20of%20multiple%20training
6. Kinza Yasar, What is machine learning and how does it work? In-depth guide. DEFINITION Neural Network, https://www.techtarget.com/searchenterpriseai/definition/neural-network
7. Neural Network example from WQU Machine Learning in Finance Module 5
8. Priya Pedamkar, Neural Network Algorithms, Updated March 18 of 2023, https://www.educba.com/neural-network-algorithms/