NEURAL NETWORKS ANALYSIS

Code used for all the neural networks analysis – Vaishnavi Panchavati

I. DIGITS DATASET

Pre-processing: The dataset is normalised before feeding it into the neural network architecture. The dataset contains all numerical values.
Testing method: K cross validation with K=10
Stopping criteria : Number of epochs(iterations) = 500
Value of alpha = 1

The table summarizes the different architectures and the performance on both training and testing data.

	Hyper-parameter variations		Model's performance	
S.No	Architecture	Lambda	Accuracy	F1-score
1.	1 hidden layer 16 neurons	0.25	.990	.958
2.	2 hidden layers 16,32 neurons	0	.990	.95
3.	3 hidden 8,8,8 neurons	0	.972	.8633
4.	3 hidden 8,8,8 neurons	0.25	.968	.8408

Choice of algorithm for the above dataset:

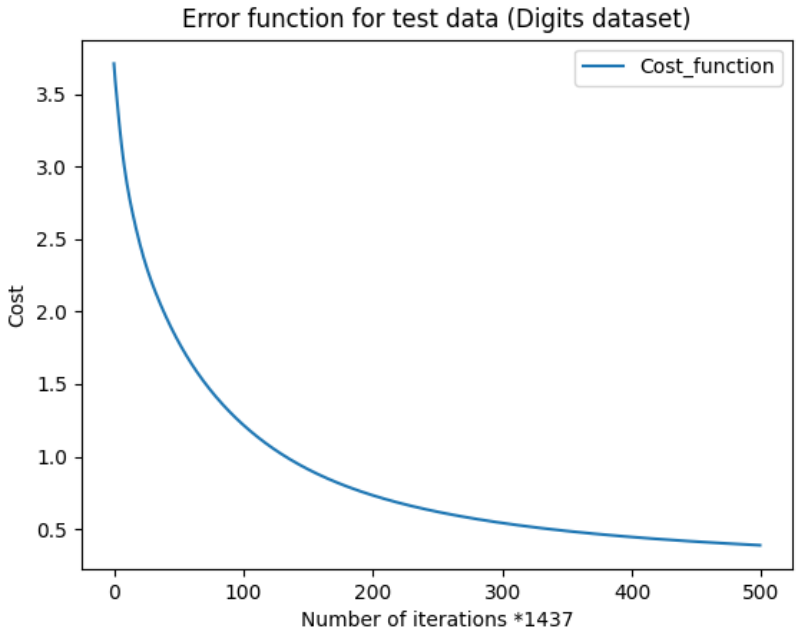
The neural networks algorithm was chosen as one of the algorithms for evaluating digits dataset due to the following reasons:

- The digits dataset consists of images representing handwritten digits. Neural networks are well-suited for capturing complex patterns and relationships in image data.
- Neural networks have shown exceptional performance on various image classification tasks. They have achieved state-of-the-art results on benchmark datasets, including handwritten digit recognition. Therefore, leveraging neural networks for the digits dataset is a natural choice to explore their potential for accurate classification.

Performance Analysis:

- The model's performance deteriorates both on the test and training data as the number of layers increase.
- The model performs best with a single hidden layer and digits dataset is a relatively straightforward classification task, as it consists of images of handwritten digits. Since the dataset is not extremely complex, ***a single-layer neural network with 16 neurons with an accuracy of .990 and F1 score of .958*** might be sufficient to capture the essential patterns and achieve good accuracy.

- Graph and explanation



The above graph is a plot of error function after every iteration performed on 80 percent samples (1437) from digits dataset and evaluated on 20 percentage test data. The plot shows that the error function decreases with increasing number of iterations .

- The decreasing trend of the error function shows that the model has fared well and is able to successfully learn the discriminative features of the digits and can generalize those learnings to correctly classify new, unseen digits.
- The error function shows a smooth decreasing behaviour on the test data as the number of iterations increase. The dataset may not be very complex and the model is able to effectively reduce the error.

II. TITANIC DATASET

Pre-processing:

- The ***'Name'*** attribute column was removed as it doesn't help in learning features and patterns during training.
- The dataset contains both numerical and categorical values.
- The categorical columns are one hot encoded and numerical columns are normalised.

Testing method: K cross validation with K=10
Stopping criteria : Number of epochs(iterations) = 500
Value of alpha = 1

The table summarizes the different architectures and the performance on both training and testing data.

	Hyper-parameter variations		Model's performance	
S.No	Architecture	Lambda	Accuracy	F1-score
1.	1 hidden layer 16 neurons	0.25	.806	.791
2.	1 hidden layers 32 neurons	0.25	.816	.802
1.	1 hidden 64 neurons	0	.813	.800
2.	2 hidden 4,16 neurons	0.25	.806	.791
3.	2 hidden 32,64 neurons	0.25	.793	.778
4.	3 hidden 4,16,32 neurons	0.25	.793	.779

Choice of algorithm for the above dataset:

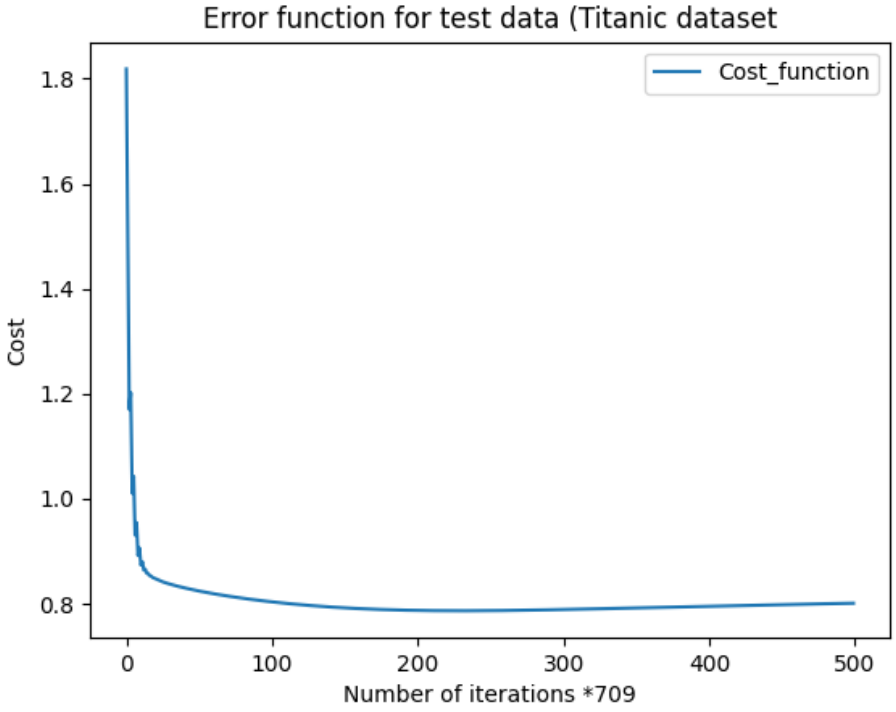
The neural networks algorithm was chosen as one of the algorithms for evaluating titanic dataset due to the following reasons:

- Neural networks excel at modelling complex, non-linear relationships between input features and target variables. The Titanic dataset may contain non-linear patterns and interactions between various features (such as age, gender, Pclass, etc.), which neural networks can effectively capture.
- Neural networks offer flexibility in terms of architecture design. You can experiment with different network architectures, layer sizes, and regularization parameter to find the best configuration for predicting survival in the Titanic dataset.

Performance Analysis:

- The model's performance slightly deteriorates as the complexity of the model increases.
- The model performs best with ***a single hidden layer and 32 neurons with an accuracy of .816 and error function of .802***.

- Graph and explanation:



The above graph is a plot of error function after every iteration performed on 80 percent samples (709) from titanic dataset and evaluated on 20 percentage test data. The plot shows that the error function decreases with increasing number of iterations .

- The graph initially shows slight jitters, this could possibly be because gradient descent updates the model's parameters based on the gradients of the error function. In the initial stages, when the model parameters are far from the optimal values, the gradients can vary significantly, leading to jittery updates.
- The graph eventually converges to 0.8.This could be because the model is able to effectively learn the different features as the number of iterations increase and weights updated to reduce error.

III. LOAN DATASET

Pre-processing:

- The ***'Loan_ID'*** attribute column was removed as it doesn't help in learning features and patterns during training.
- The dataset contains both numerical and categorical values.
- The categorical columns are one hot encoded and numerical columns are normalised.

Testing method: K cross validation with K=10
Stopping criteria : Number of epochs(iterations) = 500
Value of alpha = 1

The table summarizes the different architectures and the performance on both training and testing data.

	Hyper-parameter variations		Model's performance	
S.No	Architecture	Lambda	Accuracy	F1-score
1.	1 hidden layer 4 neurons	0.25	.800	.75
2.	1 hidden layers 32 neurons	0	.79	.742
3.	1 hidden 32 neurons	0.25	.775	.71
4.	2 hidden 4,16 neurons	0	.785	.731
5.	2 hidden 4,16 neurons	0.25	.806	.7632
6.	3 hidden 4,16,32 neurons	0.25	.787	.732

Choice of algorithm for the above dataset:

The neural networks algorithm was chosen as one of the algorithms for evaluating titanic dataset due to the following reasons:

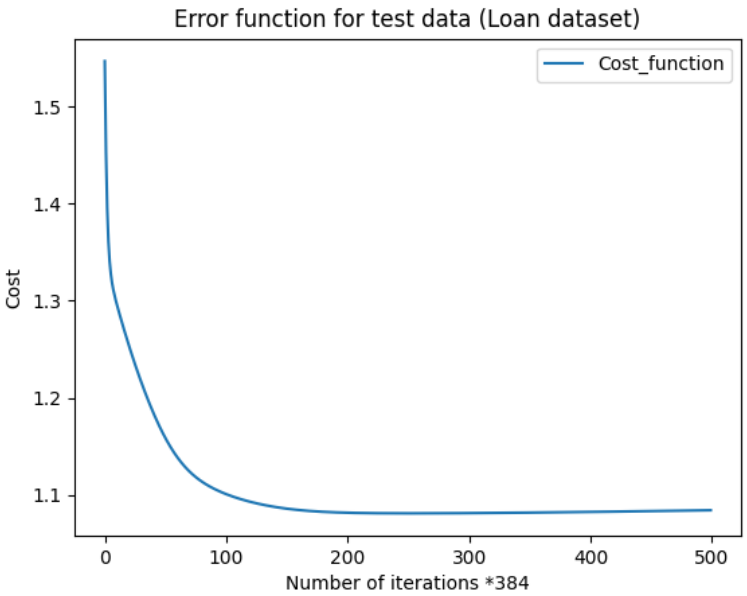
- Neural networks excel at modelling complex, non-linear relationships between input features and target variables. The loan dataset may contain non-linear patterns and interactions between various features (such as applicant_income,loan_amount_term etc.), which neural networks can effectively capture.

- Neural networks can capture interactions between features, which is important in loan datasets where the impact of one feature may depend on the values of other features. Neural networks can automatically capture and learn interactions.

Performance Analysis:

- The model's performance gets better as the as the complexity of the model increases to a certain extent.
- The model performs best with ***a 2 hidden layers with 4,16 neurons with an accuracy of .806 and error function of .763***.

- Graph and explanation:



The above graph is a plot of error function after every iteration performed on 80 percent samples (384) from loan dataset and evaluated on 20 percentage test data. The plot shows that the error function decreases with increasing number of iterations .

- The graph shows a monotonically decreasing behaviour for the best architectures chosen. This means that the model is able to learn different input features but the error value converges at about 1.1.This means that the model is not able to completely interpret the relation between different columns.
- Furthermore the dataset is complex as even with 3 layers, the error is not entirely decreased and the accuracy and f1 score are also not high.

IV. PARKINSON'S DATASET

Pre-processing: The dataset contains only numerical values and the values are normalized.
Testing method: K cross validation with K=10
Stopping criteria : Number of epochs(iterations) = 500
Value of alpha = 1

The table summarizes the different architectures and the performance on both training and testing data.

	Hyper-parameter variations		Model's performance	
S.No	Architecture	Lambda	Accuracy	F1-score
1.	1 hidden layer 4 neurons	0.25	.851	.787
2.	1 hidden layers 16 neurons	0	.841	.774
3.	2 hidden 32,32 neurons	0	.902	.867
4.	2 hidden 64,128 neurons	0	.917	.893
5.	2 hidden 128,128 neurons	0	.912	.883
6.	2 hidden 256,128 neurons	0	.917	.885
7.	3 hidden 32,64,128 neurons	0	.897	.856

Choice of algorithm for the above dataset:

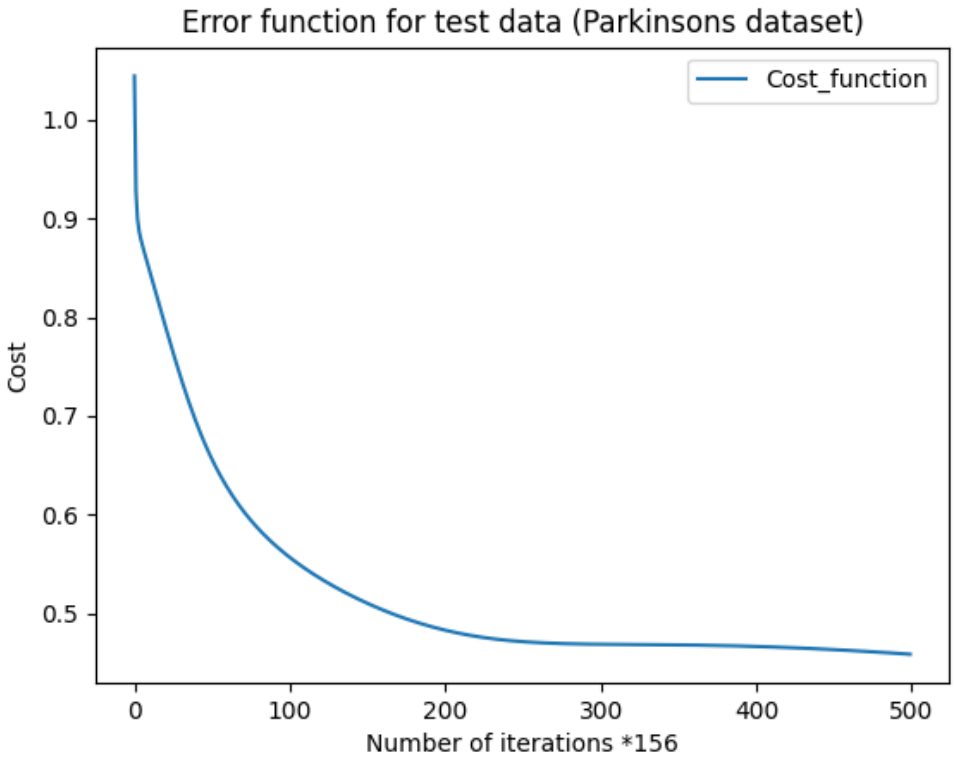
The neural networks algorithm was chosen as one of the algorithms for evaluating titanic dataset due to the following reasons:

- Neural networks, especially deep learning architectures with multiple hidden layers, have the ability to learn hierarchical representations of data. They can discover complex, abstract features and representations from the numerical inputs, potentially uncovering deeper insights into the Parkinson's disease patterns and improving the model's performance.
- Neural networks can capture interactions between features, which is important in Parkinson's datasets where the impact of one feature may depend on the values of other features. Neural networks can automatically capture and learn interactions.

Performance Analysis:

- The model's performance gets better as the as the complexity of the model increases to a certain extent.
- The model performs best with ***a 2 hidden layers with 64,128 neurons with an accuracy of .917 and error function of .893***.
- This shows that deeper networks with more neurons per layer are required for capturing patterns in Parkinson's dataset.

- Graph and performance



The above graph is a plot of error function after every iteration performed on 80 percent samples (156) from Parkinson's dataset and evaluated on 20 percentage test data. The plot shows that the error function decreases with increasing number of iterations .

- The graph shows a smooth monotonically decreasing behaviour. The model is performing well on this data as its mostly able to capture relationships between different features.
- The accuracy and f1 score are an indication of the model doing pretty well in predicting the output.

RANDOM FOREST ANALYSIS

Code used for all the random forest data – Priyanka Virupakshappa Devoor

For all the random forest analysis hyper-parameters tested are maximum depth and number of trees in the ensemble

I. DIGITS DATASET

The dataset contains all numerical values.
Testing method: K cross validation with K=10

	Hyper-parameter variations		Model's performance	
S.No	Number of trees	Stopping criterion [maximum depth]	Accuracy	F1-score
1.	5	7	0.869	0.868
2.	10	7	0.924	0.923
3.	20	7	0.948	0.948
4.	30	7	0.956	0.955
5.	10	12	0.949	0.948
6.	30	12	0.970	0.969
7.	50	12	0.974	0.974
8.	10	15	0.950	0.949
9.	30	15	0.965	0.964
10.	50	15	0.972	0.974

Analysis for different hyper parameter settings

Choice of algorithm for the above dataset:

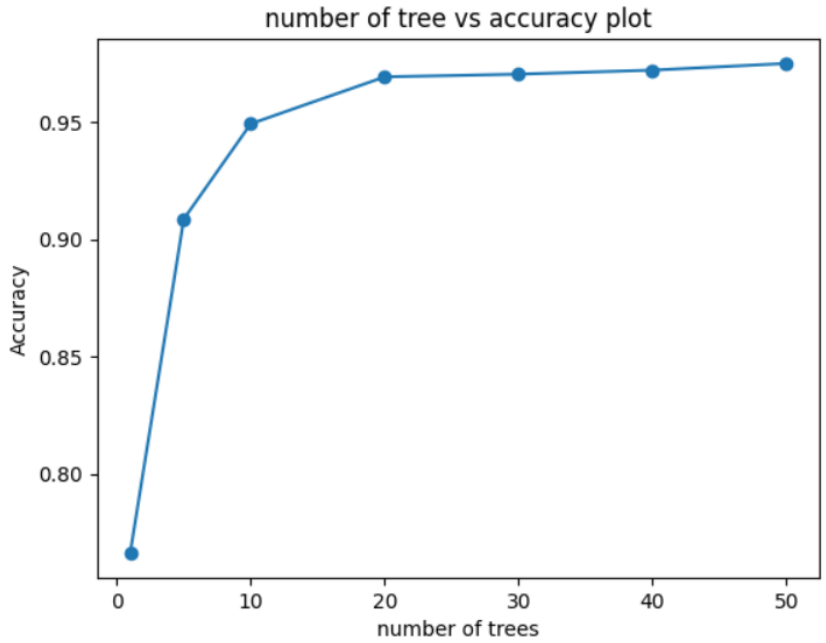
Random forest algorithm was chosen as one of the algorithms for evaluating digits dataset due to the following reasons:

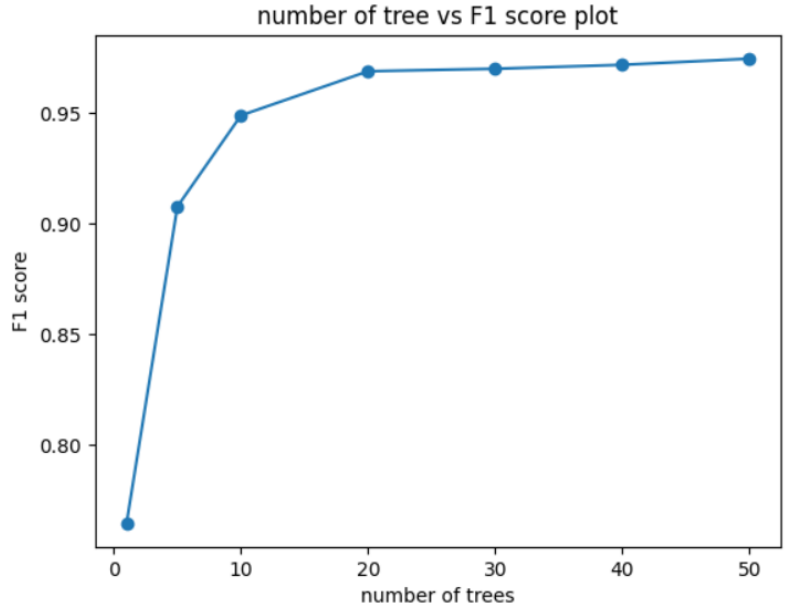
- The digits dataset consists of 64 attributes with ~1700 instances. Random forests can effectively handle high dimensional data making it suitable for image classification tasks.
- Random Forest is known to be robust to noisy data and outliers. Due to the ensemble nature of the Random Forest, variances in handwriting styles, noise, or outliers in the digit pictures may not have a significant impact on the predictions made by the algorithm.

Performance Analysis:

- It can be observed that when the maximum depth was increased from 7 to 12 for training the accuracy and F1 score increased, on further increase of the depth from 12 to 15 there were no much difference.
- Hyper-parameter setting which gave best model's performance with ***accuracy of 0.974***
  Number of trees – 50
  Maximum depth – 12

Graphs for depth = 12 and ntrees = 50





In random Forest ensemble of decision trees, are used where each tree is built using a random subset of features and a random subset of instances from the training data. This randomness helps to reduce overfitting by introducing diversity in the individual trees. Hence, as the number of trees in the ensemble increases the model is able to generalize thus giving its best accuracy and performance with 50 trees.

II. TITANIC DATASET

Pre-processing:

- The ***'Name'*** attribute column was removed as it doesn't help in learning features and patterns during training.
- The dataset contains both numerical and categorical values.

Testing method: K cross validation with K=10

	Hyper-parameter variations		Model's performance	
S.No	Number of trees	Stopping criterion [maximum depth]	Accuracy	F1-score
1.	5	7	0.809	0.79
2.	10	7	0.819	0.802
3.	20	7	0.815	0.798
4.	10	10	0.811	0.794
5.	20	10	0.822	0.81
6.	30	10	0.824	0.81
7.	10	15	0.801	0.786
8.	20	15	0.794	0.778
9.	30	15	0.807	0.798

Analysis for different hyper parameter settings

Choice of algorithm for the above dataset:

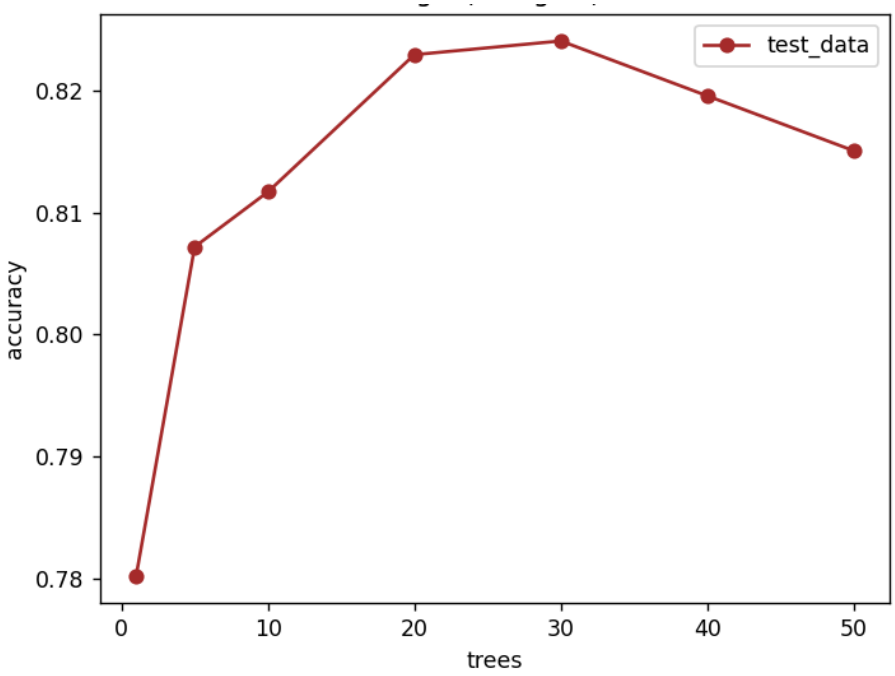
Random forest algorithm was chosen as one of the algorithms for evaluating titanic dataset due to the following reasons:

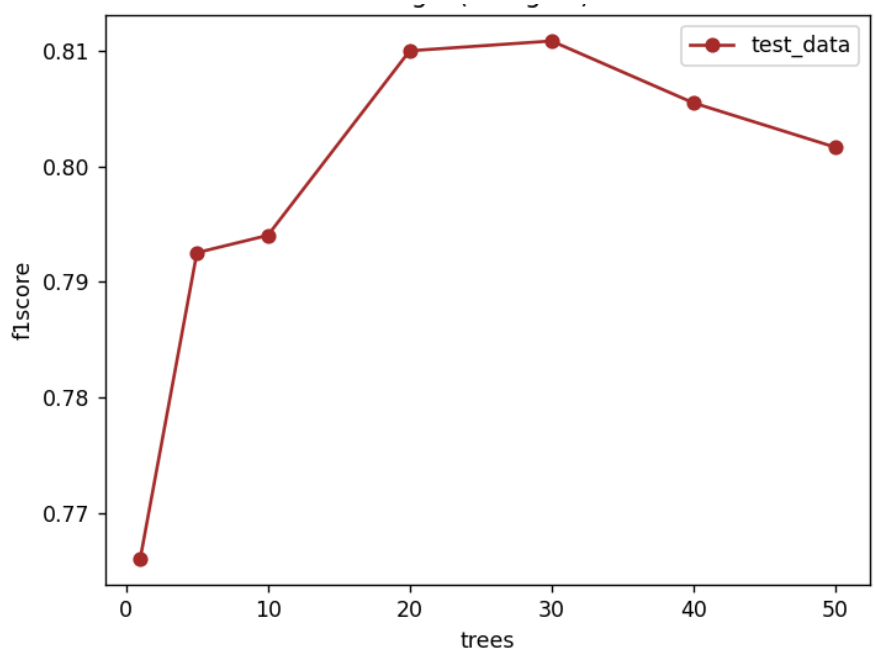
- The Titanic dataset includes both category and numerical information, such as passenger class (Pclass), gender, age, and fee. Random Forest can easily handle categorical and numerical data, making it well-suited for this dataset without the need for substantial feature engineering or data transformation.
- Random Forest is resistant to missing data and can manage noisy features without affecting its performance significantly. Random Forest can create predictions based on the available information in each tree, and its ensemble nature helps to reduce the influence of noisy or missing data points.

Performance Analysis:

- It can be observed that when the maximum depth was changed 7 to 10 model's performance enhanced whereas the performance reduces when the depth was increased to 15.
- Hyper-parameter setting which gave best model's performance with ***accuracy of 0.824***
  Number of trees – 30
  Maximum depth – 10

Graphs for depth = 10 and ntrees = 30





In random Forest ensemble of decision trees, are used where each tree is built using a random subset of features and a random subset of instances from the training data. This randomness helps to reduce overfitting by introducing diversity in the individual trees. However, as the number of trees increases, the model may become too complex and start to overfit the training data. This might be one of the reasons of why the performance reduced after ntrees=30

III. LOAN DATASET

Pre-processing:

- The ***'Loan_ID'*** attribute column was removed as it doesn't help in learning features and patterns during training.
- The dataset contains both numerical and categorical values.

Testing method: K cross validation with K=10

	Hyper-parameter variations		Model's performance	
S.No	Number of trees	Stopping criterion [maximum depth]	Accuracy	F1-score
1.	10	5	0.793	0.70
2.	30	5	0.806	0.844
3.	40	5	0.808	0.846
4.	10	8	0.766	0.698
5.	20	8	0.781	0.700
6.	30	8	0.793	0.714
7.	20	7	0.8	0.72
8.	30	7	0.806	0.73
9.	40	7	0.802	0.724

Analysis for different hyper parameter settings

Choice of algorithm for the above dataset:

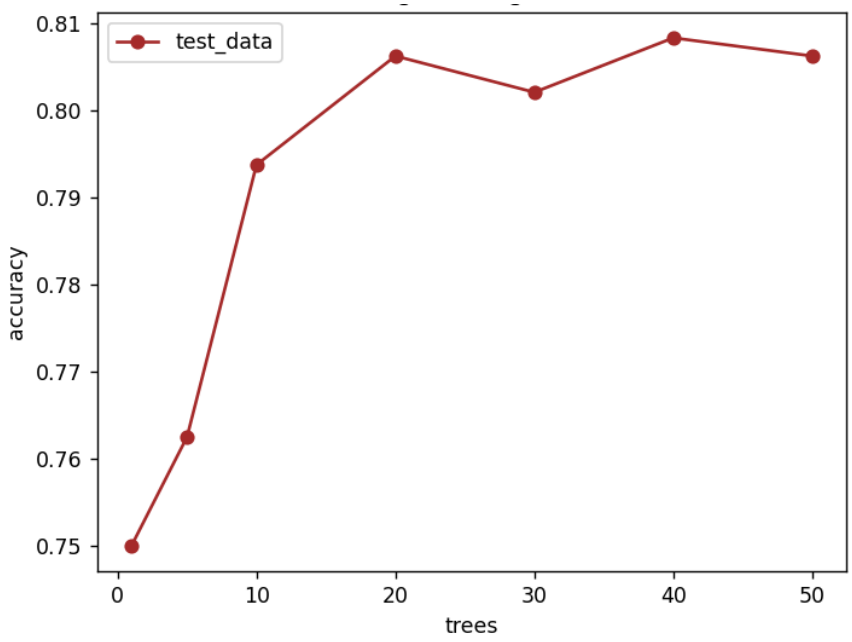
Random forest algorithm was chosen as one of the algorithms for evaluating loan dataset due to the following reasons:

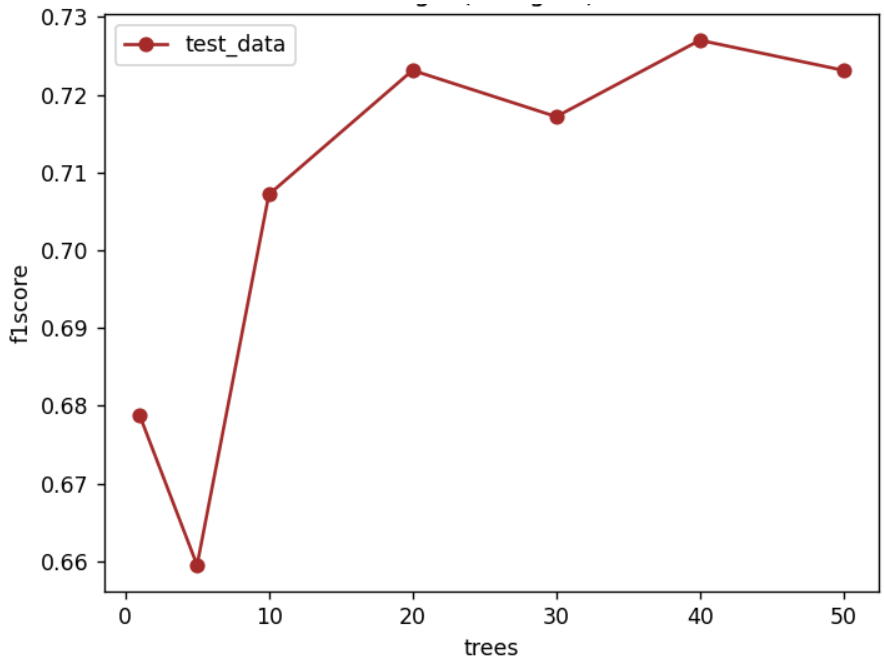
- The loan dataset includes both category and numerical information. Random Forest can easily handle categorical and numerical data, making it well-suited for this dataset without the need for substantial feature engineering or data transformation.
- Random Forest is resistant to missing data and can manage noisy features without affecting its performance significantly. Random Forest can create predictions based on the available information in each tree, and its ensemble nature helps to reduce the influence of noisy or missing data points.

Performance Analysis:

- It can be observed that when the maximum depth was changed from 5 to 7 and 7 to 8 there were no much difference in the performance.
- Hyper-parameter setting which gave best model's performance with ***accuracy of 0.808***
  Number of trees – 40
  Maximum depth – 5

Graphs for depth = 5 and ntrees = 40





In random Forest ensemble of decision trees, are used where each tree is built using a random subset of features and a random subset of instances from the training data. This randomness helps to reduce overfitting by introducing diversity in the individual trees. However, as the number of trees increases, the model may become too complex and start to overfit the training data. These might be one of the reasons for why there are performance glitches seen trees=30 and 50.

IV. PARKINSONS DATASET

Pre-processing:
The dataset contains only numerical values and the values are normalized.
Testing method: K cross validation with K=10

	Hyper-parameter variations		Model's performance	
S.No	Number of trees	Stopping criterion [maximum depth]	Accuracy	F1-score
1.	10	7	0.927	0.886
2.	20	7	0.933	0.889
3.	30	7	0.922	0.866
4.	5	15	0.911	0.866
5.	20	15	0.916	0.861
6.	30	15	0.933	0.894
7.	5	25	0.905	0.859
8.	20	25	0.933	0.88
9.	30	25	0.944	0.893

Analysis for different hyper parameter settings

Choice of algorithm for the above dataset:

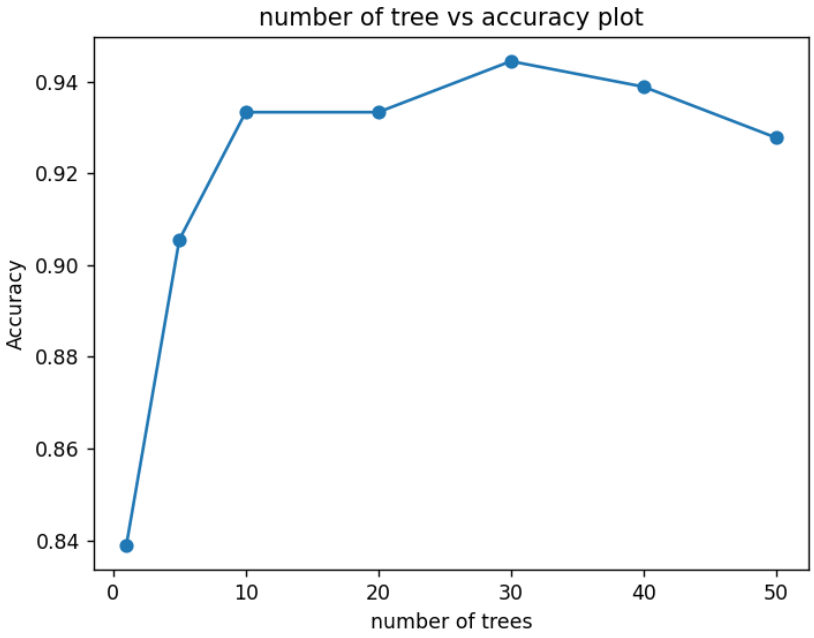
Random forest algorithm was chosen as one of the algorithms for evaluating parkinsons dataset due to the following reasons:

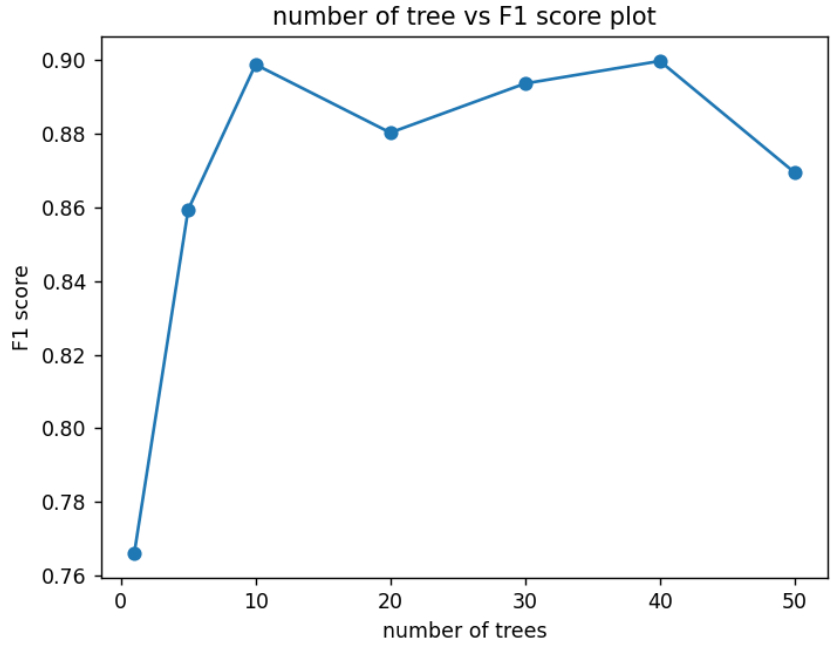
- Random Forest can model complex non-linear interactions between input data and goal variables. Parkinson's disease is a complex neurological condition with nonlinear correlations between numerical variables and disease outcome. Random Forest is capable of detecting and modeling non-linear correlations, potentially boosting prediction accuracy.

Performance Analysis:

- It can be observed that when the maximum depth was changed from 7 to 15 and then to 25 the model's performance increased.
- Hyper-parameter setting which gave best model's performance with ***accuracy of 0.944***
  Number of trees – 30
  Maximum depth – 25

Graphs for depth = 25 and ntrees = 30





In random Forest ensemble of decision trees, are used where each tree is built using a random subset of features and a random subset of instances from the training data. This randomness helps to reduce overfitting by introducing diversity in the individual trees. However, as the number of trees increases, the model may become too complex and start to overfit the training data. This might be one of the reasons of why the performance reduced after ntrees=40.

EXTRA CREDIT #1

KNN ALGORITHM ANALYSIS

I. DIGITS DATASET

Pre-processing: The dataset is normalised before feeding it into the neural network architecture.The dataset contains all numerical values.

Testing method:

- Vary value of K from 1 to about 51 with increase by 2.
- The model is evaluated using K fold cross validation.

The table summarizes the different architectures and the performance testing data.

	Hyperparameter	Model performance	
S.No	K	Accuracy	F1score
1.	1	.982	.945
2.	3	.985	.956
3.	5	.992	.958
4.	7	.986	.957
5.	9	.9859	.946
6.	11	.9845	.942
7.	13	.98345	.931

Choice of algorithm for the above dataset:

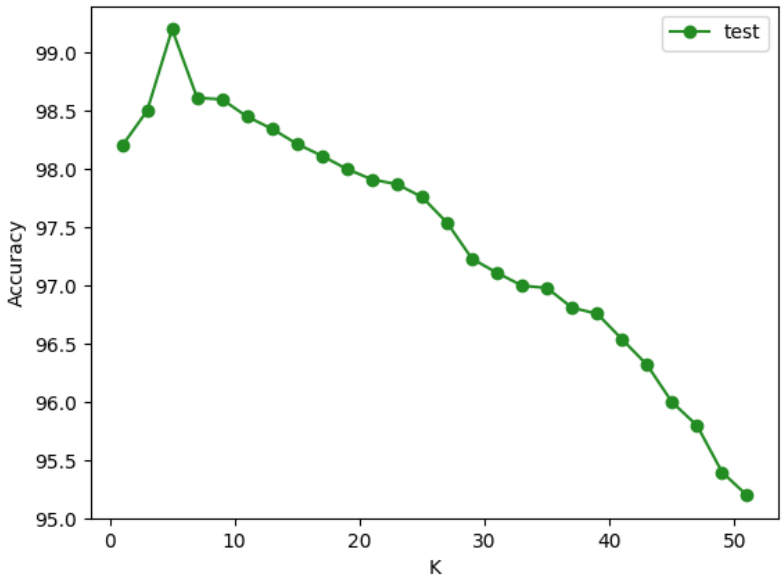
The KNN algorithm was chosen as one of the algorithms for evaluating digits dataset due to the following reason:

- k-NN defines decision boundaries based on the nearest neighbours of each data point. In the Digits dataset, digits of the same class tend to have similar pixel patterns, resulting in localized clusters. k-NN can effectively separate these clusters by considering the neighbours in the local regions, making it suitable for capturing the complex decision boundaries present in the dataset.

Performance Analysis:

- The model's performance initially increases with increasing values of K but drops after a certain value of K on the test data.
- The model performs best when ***K = 5 with an accuracy if .992 and F1score of .9587.***

- Graph and explanation



The above graph is a plot of accuracy with varying values of K from 1 to 51.

- The initial low values is because the model is overfitting on the training data. As the value of K increases, the accuracy increases as the model is able to capture the characteristics of the data effectively.
- The further drop in accuracy indicates that the model is underfitting as the label for an input depends on the majority label.

II. TITANIC DATASET

Pre-processing:

- The ***'Name'*** attribute column was removed as it doesn't help in learning features and patterns during training.
- The dataset contains both numerical and categorical values.
- The categorical columns are one hot encoded and numerical columns are normalised.

Testing method: Vary value of K from 1 to about 51 with increase by 2.The model is evaluated using K fold cross validation.

The table summarizes the different architectures and the performance on testing data

	Hyperparameter	Model performance	
S.No	K	Accuracy	F1score
1.	1	.739	.726
2.	5	.773	.758
3.	11	.779	.764
4.	19	.785	.769
5.	23	.788	.773
6.	29	.792	.776
7.	31	.793	.776
8.	39	.792	.775
9.	49	.793	.775

Choice of algorithm for the above dataset:

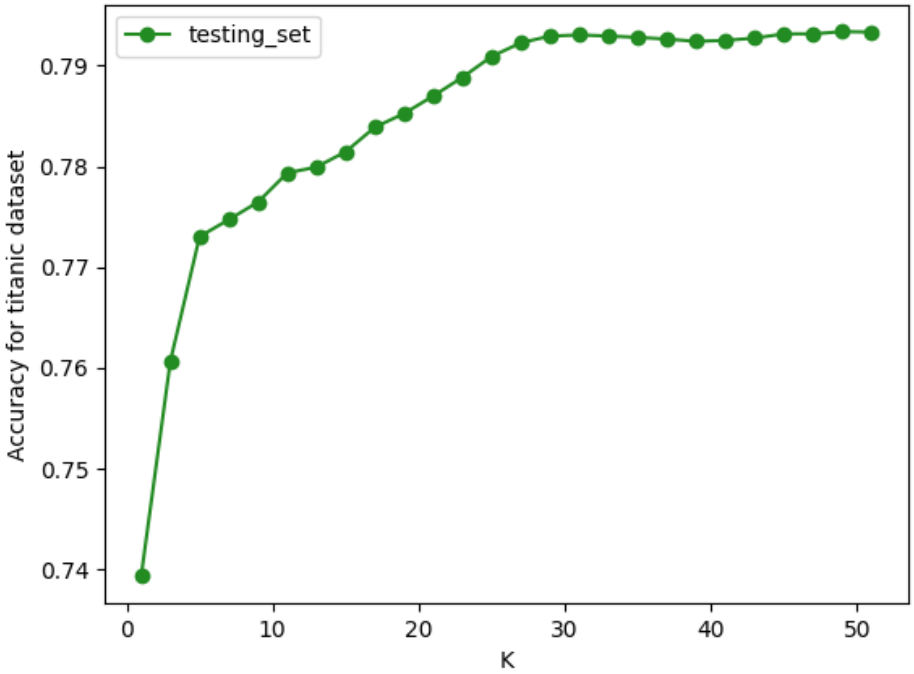
The KNN algorithm was chosen as one of the algorithms for evaluating digits dataset due to the following reason:

- The Titanic dataset may contain complex relationships between the input features and the loan approval status. Traditional linear models may not capture these non-linear relationships effectively. k-NN is a non-parametric algorithm and can handle non-linear data effectively by relying on the distances between instances in the feature space.

Performance Analysis:

- The model's performance initially increases with increasing values of K and is almost stagnant for varied values of K.
- The model performs best when ***K = 31 with an accuracy if .793 and F1score of .776.***

- Graph and explanation



The above graph is a plot of accuracy with varying values of K from 1 to 51.

- The initial low values is because the model is overfitting on the training data. As the value of K increases, the accuracy increases as the model is able to capture the characteristics of the data effectively.
- The model is complicated since it takes a large value of ***K = 31*** to get the best accuracy.
- The further stagnancy of the graph indicates that the best accuracy that can be achieved using KNN is about 0.79.

III. LOAN DATASET

Pre-processing:

- The ***'Loan_ID'*** attribute column was removed as it doesn't help in learning features and patterns during training.
- The dataset contains both numerical and categorical values.
- The categorical columns are one hot encoded and numerical columns are normalised.

Testing method: Vary value of K from 1 to about 51 with increase by 2.The model is evaluated using K fold cross validation.

The table summarizes the different architectures and the performance on testing data

	Hyperparameter	Model performance	
S.No	K	Accuracy	F1score
1.	1	.71	.66
2.	5	.72	.65
3.	11	.745	.647
4.	15	.746	.641
5.	19	.744	.632
6.	29	.742	.615
7.	33	.740	.607
8.	39	.737	.596
9.	51	.733	.578

Choice of algorithm for the above dataset:

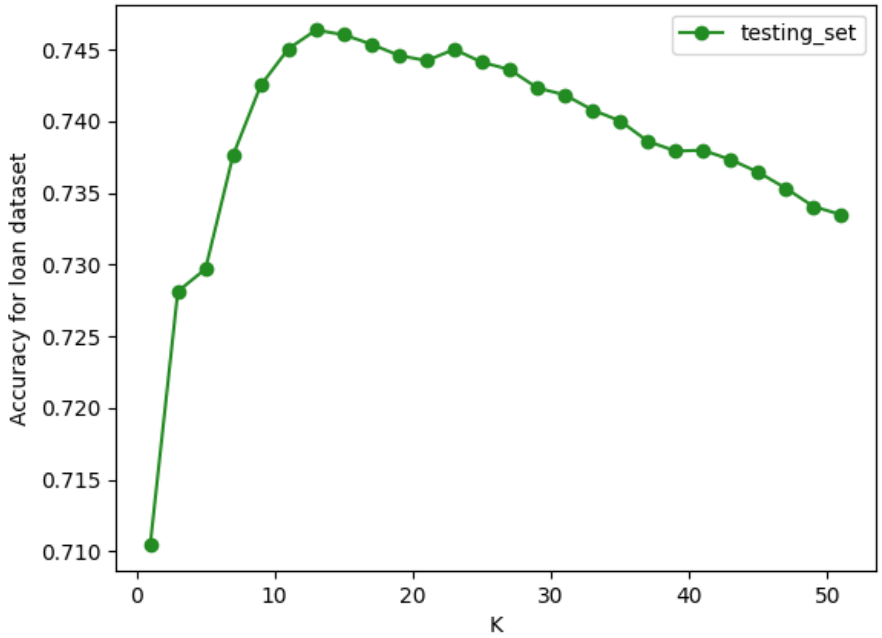
The KNN algorithm was chosen as one of the algorithms for evaluating digits dataset due to the following reason:

- The Loan dataset may contain complex relationships between the input features and the loan approval status. Traditional linear models may not capture these non-linear relationships effectively. k-NN is a non-parametric algorithm and can handle non-linear data effectively by relying on the distances between instances in the feature space.

Performance Analysis:

- The model's performance initially increases with increasing values of K and slightly drops with increasing values of K.
- The model performs best when ***K = 15 with an accuracy if .746 and F1score of .641.***
- This shows that the maximum accuracy KNN can get is very less.

- Graph and explanation



The above graph is a plot of accuracy with varying values of K from 1 to 51.

- The initial low values is because the model is overfitting on the training data. As the value of K increases, the accuracy increases as the model is able to capture the characteristics of the data effectively.
- The data is extremely hard for KNN to classify it. The best it can do is to get to an accuracy of .745.

IV. PARKINSON'S DATASET

Pre-processing:

- The dataset contains only numerical values which are normalized before feeding it into the algorithm.
- Testing method: Vary value of K from 1 to about 51 with increase by 2.The model is evaluated using K fold cross validation.

The table summarizes the different architectures and the performance on testing data

	Hyperparameter	Model performance	
S.No	K	Accuracy	F1score
1.	1	.964	.952
2.	5	.947	.929
3.	9	.932	.906
4.	13	.914	.878
5.	17	.896	.848
6.	23	.878	.813
7.	27	.871	.797
8.	33	.865	.782
9.	51	.845	.735

Choice of algorithm for the above dataset:

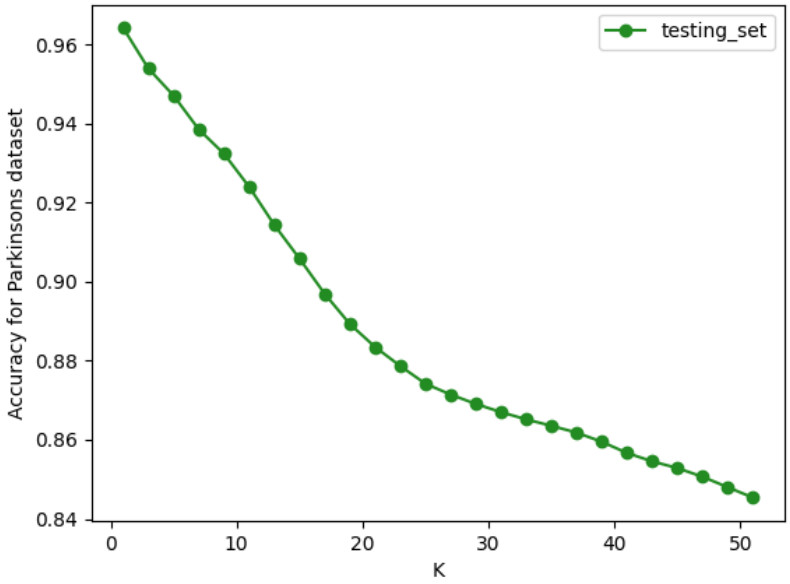
The KNN algorithm was chosen as one of the algorithms for evaluating digits dataset due to the following reason:

- k-NN classifies data points based on the majority class of their nearest neighbours. In the context of the Parkinson's dataset, similar instances in terms of their feature values may share similar disease stages or characteristics. By considering the nearest neighbours, k-NN can use this similarity and make predictions based on the class distribution of nearby instances.

Performance Analysis:

- The model's performance drops with increasing values of K.
- The model performs best when ***K = 1 with an accuracy if .964 and F1score of .952.***
- This shows that the model is performing extremely well.

- Graph and explanation



The above graph is a plot of accuracy with varying values of K from 1 to 51.

- The accuracy is very high in the beginning when the value of k is very low. As the value of k increases, the accuracy falls. The value of 1 is enough to capture the data
- The accuracy is falling because as the number of neighbours increase probably because as the neighbours increase, the predicted label becomes the majority label of the data.

COMBINED ANALYSIS OF ALL 3 ALGORITHM

	Digits dataset		Titanic dataset		Loan dataset		Parkinson's dataset	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
Neural Network	0.990	0.958	0.816	0.802	0.806	0.7632	0.917	0.893
Random Forest	0.974	0.974	0.824	0.81	0.808	0.846	0.944	0.893
K-nn	0.992	0.958	0.793	0.776	0.746	0.641	0.964	0.952