\documentclass{article}

\usepackage[utf8]{inputenc}

\usepackage{geometry}

\usepackage{graphicx}

\usepackage{amsmath}

\usepackage{amsfonts}

\usepackage{hyperref}

\usepackage{caption}

\usepackage{booktabs}

\usepackage[table]{xcolor}

\usepackage{graphicx}

\usepackage[table]{xcolor}

% \usepackage[table,xcdraw]{xcolor}

\usepackage{colortbl}

%removes indent

\setlength{\parindent}{0pt}

% Document Setup

\title{Exploring Machine Learning Model Performance across Diverse Datasets: A Comparative Analysis}

\author{Yan Mazheika \and Polina Petrova}

\date{\today}

\begin{document}

\maketitle

\section\*{Introduction}

This report provides an analysis of using various machine learning algorithms on various datasets.

The goal is to identify the best-use scenarios of model variants given the nature of input data.

In this report, the authors analyze the

\begin{itemize}

\item Decision Tree, \textit{*provided by Polina Petrova*}

\item k Nearest Neighbors, \textit{*provided by Polina Petrova*}

\item Neural Network, \textit{*provided by Yan Mazheika*}

\item Random Forrest, \textit{*provided by Yan Mazheika*}

\end{itemize}

against the handwritten digits, titanic survival, loan eligibility, and Parkinson's classification datasets.

\\

It is our aim to explain the nature of the data, our models, and the performance of these classifiers. Throughout our report,

we justify our algorithm choice for the given dataset and our choice of hyper-parameters for the tuning of the algorithm. These insights

are for the reader's benefit; we also aim to provide insight into how to solve novel machine problems, which algorithms may work best, and how

to adjust their hyper-parameters for optimal performance.

\subsection\*{Approach}

The authors have chosen to coordinate on every dataset. For the first two algorithms of every dataset, one of them comes from \textit{*Yan Mazheika*}

while the other comes from \textit{*Polina Petrova*}. These algorithms may have been modified to handle a new type of data

but their fundamental logic stays consistent from previous use cases.

\\

We use cross-validation of k=10 folds when testing the performance of our models (except for kNN). We also transform the data into the [0,1] range when using the neural network or kNN classifiers.

\subsection\*{Performance Overview}

% Please add the following required packages to your document preamble:

% \usepackage[table,xcdraw]{xcolor}

% Beamer presentation requires \usepackage{colortbl} instead of \usepackage[table,xcdraw]{xcolor}

\newgeometry{left=1cm,right=1cm}

\begin{table}[h]

\begin{tabular}{l|cc|cc|cc|cc}

Dataset: & \multicolumn{2}{c|}{Digits} & \multicolumn{2}{c|}{Titanic} & \multicolumn{2}{c|}{Loan} & \multicolumn{2}{c}{Parkinson's} \\ \hline

& Accuracy & F1-Score & Accuracy & F1-Score & Accuracy & F1-Score & Accuracy & F1-Score \\ \cline{2-9}

k-NN & .8698 & .9298 & - & - & - & - & 0.7526 & 0.8557 \\

Decision Tree & - & - & .7023 & .8234 & .5354 & \cellcolor[HTML]{C0C0C0}.6860 & - & - \\

Random Forrest & - & - & - & - & \cellcolor[HTML]{C0C0C0}.7354 & .6650 & - & - \\

Neural Network & \cellcolor[HTML]{C0C0C0}.9962 & \cellcolor[HTML]{C0C0C0}.9809 & \cellcolor[HTML]{C0C0C0}.9663 & \cellcolor[HTML]{C0C0C0}.9652 & - & - & \cellcolor[HTML]{C0C0C0}.9226 & \cellcolor[HTML]{C0C0C0}.8940 \\

Additional Algorithm & - & - & - & - & - & - & - & -

\end{tabular}

\end{table}

\restoregeometry

\newpage

\section\*{Digits Dataset}

The digits dataset come's from the sci-kit learn library, available in Python. We have chosen to analyze the performance of a neural network classifier and the k-Nearest Neighbors classifier. Again, we mention

that the neural network tuning was performed by \textit{*Yan Mazheika*} while the kNN tuning was done by \textit{*Polina Petrova*}.

\\

We chose a neural network as one of our algorithms for this dataset because of the algorithm's ability to

handle complex patterns and identify non-linear patterns in these data. Anecdotally, neural networks have shown

impressive results in image classification tasks, making them a top contender for handwritten digit recognition, the focus of this dataset.

\\

On the other hand, the k-NN algorithm is a simple yet effective classifier that makes predictions based on the closeness of past training instances in the feature space. We figured that similar pixel activations in the handwritten digits

would translate into less distance in the 64-dimensional feature space. A concern we had when choosing this algorithm is precision loss; kNN performs poorly with a large number of features, which is 64 in our case. The distance measurement between two instances

converges to zero as we add more features and normalize them. However, this wasn't a major problem in our analysis.

\newgeometry{left=1cm,right=1cm}

\subsection\*{Performance Metrics}

%we can adjust for three models later

\begin{center}

\texttt{Neural Network}

\includegraphics\*[width=0.6\textwidth]{./src/figures/Digits-test-cost.png}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\begin{tabular}{lc}

\toprule

\multicolumn{2}{c}{Neural Network} \\

\midrule

Learning Rate $\alpha$ & 0.0500 \\

Regularization $\lambda$ & 0.0100 \\

Architecture & [64, 32, 32, 10] \\

Mean Accuracy & 0.9962 \\

Mean F1-score & 0.9809 \\

Mean Test Cost & 59.3100 \\

\bottomrule

\end{tabular}

\end{minipage}

\hfill

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=1\textwidth]{./src/figures/Digits\_train\_cost.png}

\captionof\*{figure}{\textit{*Test cost over 179,700 instances*}}

\vfill

\end{minipage}

\newpage

\begin{center}

\texttt{k-NN Algorithm}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=\textwidth]{./src/figures/Accuracy Digits.png}

\includegraphics\*[width=\textwidth]{./src/figures/F-Score Digits.png}

\begin{tabular}{lc}

\toprule

\multicolumn{2}{c}{k-NN} \\

\midrule

Neighbors $k$ & 51 \\

Mean Accuracy & 0.8698 \\

Mean F1-score & 0.9298 \\

\bottomrule

\end{tabular}

\captionof\*{figure}{\textit{*Metrics for Preferred Model Variant*}}

\end{minipage}

\begin{minipage}{0.49\textwidth}

\centering

\vfill

\begin{tabular}{lrrr}

\toprule

$k$ & Test Accuracy & Test F-Score \\

\midrule

1 & {0.7291} & {0.8413} \\

3 & {0.7615} & {0.8625} \\

5 & {0.7615} & {0.8629} \\

7 & {0.7799} & {0.8748} \\

9 & {0.7872} & {0.8797} \\

11 & {0.7911} & {0.8821} \\

13 & {0.7939} & {0.8836} \\

15 & {0.8000} & {0.8875} \\

17 & {0.8073} & {0.8921} \\

19 & {0.8089} & {0.8932} \\

21 & {0.8184} & {0.8991} \\

23 & {0.8235} & {0.9022} \\

25 & {0.8307} & {0.9066} \\

27 & {0.8302} & {0.9062} \\

29 & {0.8330} & {0.9079} \\

31 & {0.8380} & {0.9111} \\

33 & {0.8397} & {0.9121} \\

35 & {0.8441} & {0.9148} \\

37 & {0.8503} & {0.9184} \\

39 & {0.8542} & {0.9208} \\

41 & {0.8547} & {0.9209} \\

43 & {0.8570} & {0.9223} \\

45 & {0.8615} & {0.9249} \\

47 & {0.8670} & {0.9281} \\

49 & {0.8682} & {0.9287} \\

51 & {0.8698} & {0.9298} \\

\bottomrule

\end{tabular}

\captionof\*{figure}{\textit{*Performance Metrics over various k*}}

\vfill

\end{minipage}

\restoregeometry

\subsection\*{Tuning Hyper-parameters}

For the neural network, we've chosen an $\alpha$ of 0.05 and a regularization constant of $\lambda=0.01$. At first, with a $\lambda=0$ and $\alpha=0.50$, the model had good performance but a quick convergence in the training curve.

We found increased performance and a slower convergence by decreasing $\alpha$ and increasing $\lambda$ to a final accuracy of 99.62\%. This indicates a misclassification of 6 total instances out of 1,797 total.

The architecture was chosen because it had the lowest accuracy variance between each fold out of the models tested.

\\

In our implementation of the k-NN algorithm, we tested different (integer) values of $k$ in the range [1, 52) for an ideal hyperparameter.

We observed an initial jump in the evaluation metrics between $k$ = 1 and $k$ = 2 and a high variance.

This points to the algorithm being sensitive to outliers at very small values of $k$.

As the value of $k$ increases, the values of the evaluation metrics increase, as well, and stabilise by $k$ = 51 at 86.98\% for accuracy and 92.98\% for F1-score.

Increasing this hyperparameter value also led to smaller variance, which is helpful when dealing with noisy datasets.

In witnessing this convergence, we have chosen a final hyperparameter value for the k-NN algorithm to be $k$ = 51.

With a larger $k$ value and smoother decision boundary, we mitigate the chance of overfitting our model to the data.

\subsection\*{Analysis}

Reporting on the final runs of the neural network and k-NN algorithms on the digits dataset, both models performed well on the testing instances.

However, testing on the neural network resulted in higher accuracy and F1-score values as opposed to when testing on k-NN.

Dealing with an 8x8 pixel image, k-NN is able to effectively measure distances in the feature space when making predictions.

Although k-NN performs well on this dataset, it struggles in distinguishing between two or more digits that look alike when handwritten.

For example, an analysis of its predictions made on an instance of class 0 may return a substantial number of predictions of class 9, which would be a factor in decreasing accuracy.

\\

The neural network acts as a well-defined alternative to this issue, as it captures more complex patterns and data variability present in handwriting.

Additionally, a higher neuron count facilitates learning these handwriting variations of numbers of the same class, making the network more reliable in predicting instances that look alike and a higher accuracy.

\newpage

\section\*{Titanic Dataset}

We pre-processed some of the titanic dataset. At first, there was a 'Sex' column which was encoded into a 'Male' column indicating a 1 if the 'Sex' of a given instance contained 'male' and a 0 if an instance contained 'female'.

There was also a 'Name' column which was removed entirely. At first, we attempted to process this by encoding every English letter into a number. This resulted in a very large data-frame that led to poor results after processing.

\\

Afterwards, we tried to one-hot encode the first word of the 'Name' column which indicated the title of a person. Because this dataset represents a time when a person's title had greater significance, we figured that there would be differences in the survival rates between prestigious and non-prestigious titles (such as 'Rev' or 'Master' versus 'Mr' or 'Miss').

We also hypothesized that persons with the title 'Rev,' which indicates religious affiliation, would be less likely to survive as we thought that their role as a religious figure would influence them to stay on the ship for longer than others. Indeed, every reverend in the dataset did not survive.

\\

Empirically, we found that this didn't work well, which led to the removal of the 'Name' feature in the dataset.

Perhaps this is because 'Pclass' contains the ticket class of a given passenger, which mitigates the role that the 'Name' column plays in indicating the socio-economic status of a passenger. Additionally,

the added dimensionality of encoding around 8 titles could have mitigated the importance of the un-altered columns.

\\

The choice of applying a neural network for this task can be justified by it's performance (speed) and ability to learn complex patterns.

It's arguably the easiest model to tune with respect to the datasets this paper presents.

\\

The choice of applying a decision tree for this task is natural as well. There are a number of more important features that a high-bias model can benefit from. These

include passenger class, ticket fare, and gender. With three main features impacting survivability, we can benefit from a decision tree by quickly sub-setting the dataset on those features.

\newpage

\subsection\*{Performance Metrics}

\begin{center}

\texttt{Neural Network}

\includegraphics\*[width=0.8\textwidth]{./src/figures/Titanic-test-cost.png}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\begin{tabular}{ll}

\toprule

\multicolumn{2}{c}{Neural Network} \\

\midrule

Learning Rate $\alpha$ & 0.0400 \\

Regularization $\lambda$ & 0.0900 \\

Architecture & [6, 4, 2] \\

Mean Accuracy & 0.9663 \\

Mean F1-score & 0.9652 \\

Mean Test Cost & 84.8500 \\

\bottomrule

\end{tabular}

\end{minipage}

\hfill

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=0.9\textwidth]{./src/figures/Titanic\_train\_cost.png}

\captionof\*{figure}{\textit{*Test cost over 87,700 instances*}}

\end{minipage}

\newgeometry{left=1cm,right=1cm}

\begin{center}

\texttt{Decision Tree}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=\textwidth]{./src/figures/Accuracy Titanic.png}

\includegraphics\*[width=\textwidth]{./src/figures/F-Score Titanic.png}

\end{minipage}

\begin{minipage}{0.49\textwidth}

\centering

\begin{tabular}{rrr}

\toprule

Maximum depth & Test Accuracy & Test F-score \\

\midrule

1 & {0.7852} & {0.8793} \\

2 & {0.6386} & {0.7774} \\

3 & {0.5898} & {0.7372} \\

4 & {0.7125} & {0.8291} \\

5 & {0.6784} & {0.7920} \\

6 & {0.6295} & {0.7540} \\

7 & {0.5227} & {0.6619} \\

8 & {0.4216} & {0.5802} \\

9 & {0.4216} & {0.5802} \\

\bottomrule

\end{tabular}

\vspace\*{30pt}

\begin{tabular}{lc}

\toprule

\multicolumn{2}{c}{Decision Tree} \\

\midrule

Maximum Depth & 4 \\

Mean Accuracy & 0.7023 \\

Mean F1-score & 0.8234 \\

\bottomrule

\end{tabular}

\captionof\*{figure}{\textit{*Metrics for Preferred Model Variant*}}

\end{minipage}

\restoregeometry

\subsection\*{Tuning Hyper-parameters}

The neural network was analyzed over various learning rates and regularization constants. We found that a learning rate of 0.04 and a regularization constant of 0.09 produced the best results.

We found that a larger-than-usual regularization constant was necessary to reducing the variance in the model's predictions. At lower $\lambda$ values, the model appeared to overfit the data, which was evident by the high variance in the training cost curve.

A smaller architecture was chosen to reduce the complexity of the model as performance suffered with the presence of more than one hidden layer.

This is likely due to the complexity of the dataset, which features rather arbitrary social factors.

\\

Conducting multiple analyses of the decision tree algorithm on the titanic dataset with differing maximum tree depths,

we have determined the preferred hyperparameter to be $max\\_depth$ = 4.

At this value, accuracy and F1-score reach a peak at 70.23\% and 82.34\%, respectively, after which both metrics decrease substantially.

Deeper trees lead to overfitting and decrease in generalisation ability,

which is evidenced by the sharp increace in variance on both accuracy and F-1 score after tree depth 4.

\subsection\*{Analysis}

Our decision tree struggled in providing reliable predictions on the titanic dataset.

This is possibly due to the fact that the dataset contained numerical attributes such as $Ticket\ Fare$ and $Age$

that may be difficult to evaluate on this model.

The decision nodes were split based on a numerical threshold that maximised information gain at each decision point.

This approach is particularly challenging given the large range of numerical values present in the data.

For example, there were passengers as old as 80 years-old on the ship, along with infants as young as 5 months-old.

Similarly, the ticket fare values ranged from 0.0 to 512, showing a wide distribution of values.

Although we employed feature normalisation, this variability in numerical attributes could have contributed to

the difficulty in finding optimal splits during the tree construction process.

Additionally, deeper trees tend to create more decision nodes based on numerical thresholds,

which may struggle to effectively differentiate between instances with similar normalised values but different original scales.

\\

On the other hand, employing a neural network on this dataset proved to be substantially more accurate in predicting instance classes.

However, we observed a concerning trend in the neural network's performance metrics.

While the accuracy on the test set was high, we witnessed a very high mean test cost, as well.

Furthermore, the extreme variance in cost as the model processed more instances

indicates that the neural network's predictions were inconsistent across different subsets of the dataset.

This may be a consequence of class imbalance in the titanic dataset.

The proportion of the class $Did\ Not\ Survive$ was 61\%,

while for the class $Survived$, the proportion was only 38\%.

Our neural network model may be prioritising minimising errors on the majority class,

leading to higher costs for misclassifying the minority class instances.

\newpage

\section\*{Loan Eligibility Dataset}

The loan eligibility dataset is a binary classification problem. We have chosen to analyze the performance of a decision tree and a random forest classifier.

\\

The choice of applying a decision tree to the problem was natural. Anecdotally, decision trees are usually taught in the context of loan eligibility problems.

This is because decision trees are based on a series of if-else statements that are both easy to interpret and prioritize the most important features.

In the context of this dataset, the presence of 11 total features makes some features more important than others, which will ideally be filtered for relevance by a decision tree.

\\

The choice of applying a random forrest was also natural. Random forests are ensembles of decision trees and aim to reduce the variance of output problems.

With this algorithm, we can ideally expect a model less prone to over-fitting.

\newgeometry{left=1cm,right=1cm}

\subsection\*{Performance Metrics}

\begin{center}

\texttt{Decision Tree}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=\textwidth]{./src/figures/Accuracy Loan.png}

\includegraphics\*[width=\textwidth]{./src/figures/F-Score Loan.png}

\centering

\begin{tabular}{lc}

\toprule

\multicolumn{2}{c}{Decision Tree} \\

\midrule

Maximum Depth & 2 \\

Mean Accuracy & 0.5354 \\

Mean F1-score & 0.6860 \\

\bottomrule

\end{tabular}

\captionof\*{figure}{\textit{*Metrics for Preferred Model Variant*}}

\end{minipage}

\begin{minipage}{0.49\textwidth}

\centering

\begin{tabular}{rrr}

\toprule

Maximum depth & Test Accuracy & Test F-score \\

\midrule

1 & {0.8083} & {0.8930} \\

2 & {0.6458} & {0.7657} \\

3 & {0.4708} & {0.6357} \\

4 & {0.3583} & {0.5139} \\

5 & {0.3104} & {0.4606} \\

6 & {0.3042} & {0.4609} \\

11 & {0.2437} & {0.3894} \\

16 & {0.2437} & {0.3894} \\

21 & {0.2437} & {0.3894} \\

26 & {0.2437} & {0.3894} \\

31 & {0.2437} & {0.3894} \\

36 & {0.2437} & {0.3894} \\

41 & {0.2437} & {0.3894} \\

46 & {0.2437} & {0.3894} \\

51 & {0.2437} & {0.3894} \\

56 & {0.2437} & {0.3894} \\

61 & {0.2437} & {0.3894} \\

66 & {0.2437} & {0.3894} \\

71 & {0.2437} & {0.3894} \\

76 & {0.2437} & {0.3894} \\

81 & {0.2437} & {0.3894} \\

86 & {0.2437} & {0.3894} \\

91 & {0.2437} & {0.3894} \\

96 & {0.2437} & {0.3894} \\

\bottomrule

\end{tabular}

\end{minipage}

\restoregeometry

\newgeometry{left=1cm,right=1cm}

\begin{center}

\texttt{Random Forrest}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=\textwidth]{./src/figures/Loan-accuracy\_final.png}

\end{minipage}

\hfill

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=\textwidth]{./src/figures/Loan-f1-score\_final.png}

\end{minipage}

\begin{center}

\begin{tabular}{rrrrr}

\toprule

accuracy & precision & recall & f1-score & ntree \\

\midrule

0.6667 & 0.5729 & 0.5493 & 0.5609 & 1 \\

0.7354 & 0.6875 & 0.6440 & 0.6650 & 2 \\

0.7271 & 0.6755 & 0.6286 & 0.6512 & 5 \\

0.7208 & 0.7436 & 0.5585 & 0.6379 & 10 \\

0.7292 & 0.8348 & 0.5627 & 0.6722 & 20 \\

0.7021 & 0.6756 & 0.5281 & 0.5928 & 30 \\

0.7104 & 0.7538 & 0.5360 & 0.6265 & 50 \\

\bottomrule

\end{tabular}

\captionof\*{figure}{\textit{*Performance metrics across various $n$-tree*}}

\end{center}

\restoregeometry

\subsection\*{Tuning Hyper-parameters}

Similar to the implementation of the decision tree model for the titanic dataset,

the performance of the model diminished with maximum depth values above 2.

Thus, we chose the hyperparameter $max\\_depth$ = 2 to represent this model.

Furthermore, we found that repeated testing with maximum depth values of 2 and 3

produced similar evaluation values.

The decision to select $max\\_depth$ = 2 stemmed from the oscillation in performance

metrics during repeated testing on $max\\_depth$ = 3,

which produced greater instability than the former hyperparameter value.

\\

[add for random forest]

\subsection\*{Analysis}

For the loan dataset, we employed the decision tree and random forest algorithms.

It is crucial to note that the random forest model used entropy as a stopping criterion,

while the decision tree model used tree depth as its stopping criterion.

This may be a factor in why the random forest classifier performed better on both accuracy and F1-score

than the decision tree classifier.

That being said, both models converged to poor local optimums,

indicating a lapse in classification ability for tree-based algorithms on this dataset.

[add why]

\newpage

\section\*{Oxford Parkinson's Disease Dataset}

\subsection\*{Performance Metrics}

\newgeometry{left=1cm,right=1cm}

\begin{center}

\texttt{k-NN Algorithm}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=\textwidth]{./src/figures/Accuracy Parkinsons.png}

\includegraphics\*[width=\textwidth]{./src/figures/F-Score Parkinsons.png}

\begin{tabular}{lc}

\toprule

\multicolumn{2}{c}{k-NN} \\

\midrule

Neighbors $k$ & 61 \\

Mean Accuracy & 0.7526 \\

Mean F1-score & 0.8557 \\

\bottomrule

\end{tabular}

\captionof\*{figure}{\textit{*Metrics for Preferred Model Variant*}}

\end{minipage}

\begin{minipage}{0.49\textwidth}

\centering

\begin{tabular}{rrr}

\toprule

k & Test Accuracy & Test F-Score \\

\midrule

1 & {0.5947} & {0.7367} \\

6 & {0.2789} & {0.4319} \\

11 & {0.2789} & {0.4319} \\

16 & {0.2474} & {0.3921} \\

21 & {0.2474} & {0.3893} \\

26 & {0.2474} & {0.3893} \\

31 & {0.2474} & {0.3893} \\

36 & {0.2474} & {0.3893} \\

41 & {0.2579} & {0.4030} \\

46 & {0.4211} & {0.5625} \\

51 & {0.6000} & {0.7320} \\

56 & {0.7211} & {0.8350} \\

61 & {0.7526} & {0.8561} \\

66 & {0.7526} & {0.8561} \\

71 & {0.7526} & {0.8561} \\

76 & {0.7526} & {0.8561} \\

81 & {0.7526} & {0.8561} \\

86 & {0.7526} & {0.8561} \\

91 & {0.7526} & {0.8561} \\

96 & {0.7526} & {0.8561} \\

101 & {0.7526} & {0.8561} \\

\bottomrule

\end{tabular}

\end{minipage}

\restoregeometry

\begin{center}

\texttt{Neural Network}

\includegraphics\*[width=0.8\textwidth]{./src/figures/Parikinson's-test-cost.png}

\end{center}

\begin{minipage}{0.49\textwidth}

\centering

\begin{tabular}{ll}

\toprule

& Neural Network \\

\midrule

Learning Rate $\alpha$ & 0.1000 \\

Regularization $\lambda$ & 0.0001 \\

Architecture & [22, 12, 2] \\

Mean Accuracy & 0.9226 \\

Mean F1-score & 0.8940 \\

Mean Test Cost & 13.4800 \\

\bottomrule

\end{tabular}

\end{minipage}

\hfill

\begin{minipage}{0.49\textwidth}

\centering

\includegraphics\*[width=0.9\textwidth]{./src/figures/Parikinson's\_train\_cost.png}

\captionof\*{figure}{\textit{*Training Cost Over 19,500 Instances*}}

\end{minipage}

\subsection\*{Tuning Hyper-parameters}

.7538 accuracy of random classifier.

\subsection\*{Analysis}

\end{document}