

EXPERT SYSTEM USING BAYESIAN NEURAL NETWORKS

*submitted in partial fulfillment of the requirements
for the degree of*

MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

SATHURI BHARATH KUMAR GOUD CS20M011

Supervisor(s)

Prof Dr. SRINIVAS PADMANABHUNI

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY TIRUPATI**

JUNE 2022

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/-data/fact/source in my submission to the best of my knowledge. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Place: Tirupati
Date: 01-06-2022

Signature
Sathuri Bharath Kumar Goud
CS20M011

BONA FIDE CERTIFICATE

This is to certify that the report titled **EXPERT SYSTEM USING BAYESIAN NEURAL NETWORKS**, submitted by **SATHURI BHARATH KUMAR GOUD**, to the Indian Institute of Technology, Tirupati, for the award of the degree of **Master of Technology**, is a bona fide record of the project work done by him under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Tirupati
Date: 01-06-2022

Dr. Srinivas Padmanabhuni
Guide
Guest Faculty
Department of Computer
Science and Engineering
IIT Tirupati - 517501

ACKNOWLEDGMENTS

I am thankful to my guide Dr. SRINIVAS PADMANABHUNI for his guidance, encouragement and continuous support for the whole duration of this project.

I would like to thank all the faculty members of the department of Computer Science and Engineering, IIT Tirupati for their constant support throughout this project and for all the courses here at IIT Tirupati and making this a memorable and successful journey and a lifelong learning experience .

I have taken the dataset for this project from UCI Dataset Repository [Dua and Graff \(2017\)](#) titled " AI4I 2020 Predictive Maintenance Dataset Data Set" provided by Stephan Matzka, School of Engineering - Technology and Life, Hochschule für Technik und Wirtschaft Berlin, 12459 Berlin, Germany.

ABSTRACT

KEYWORDS: Expert System; Bayesian Neural Networks; Streamlit; Tensorflow.

An Expert System can be defined as a knowledge-base system that acquires knowledge from its knowledge base of a specific domain and addresses user queries using its inference engine about that specific domain. In the Expert System, we have 3 main parts(Knowledge Base, Inference Engine, User Interface). It has many applications in various domains. One of the important domains is Predictive Maintenance where it helps in monitoring the health of machinery and helps us to get quick repair solutions with its knowledge and improve its lifetime. We are using Bayesian Neural Networks (these are Neural networks in which we use probabilities for weights and biases for prediction) as a part of the inference engine of the expert system which helps to overcome the uncertainties from standard Neural Networks and gives confident and dynamic predictions. This yields us even much better results and helps in effective monitoring of the machinery both by individuals and organizations.

Here, we present the usage of Bayesian Neural Networks(BNN's) for building Expert Systems. The usage of BNN's overcomes the drawback of the uncertainty of neural networks. It also makes predictions more dynamic.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	i
ABSTRACT	ii
LIST OF FIGURES	vi
LIST OF TABLES	vii
ABBREVIATIONS	viii
NOTATION	ix
1 INTRODUCTION	1
1.1 Introduction	1
1.1.1 Expert System	1
1.1.2 Neural Networks	2
1.1.3 Bayesian Neural Networks(BNN)	2
1.2 ES AND BNN	3
1.2.1 BAYESIAN NEURAL NETWORKS	3
1.2.2 EXPERT SYSTEM ARCHITECTURE	4
1.2.3 ADVANTAGES	5
1.2.4 LIMITATIONS	5
1.3 Problem Statement	6
1.4 Our Contribution	6
1.5 Organization of The Report	7
2 BACKGROUND INFORMATION	8
2.1 Expert System	8
2.2 BNN	9
3 MATERIALS AND METHODS	12

3.1	About this Chapter	12
3.2	Technologies used.	12
3.3	Dataset, Front end, Back-end	13
3.3.1	Dataset	13
3.3.2	Data Description	13
3.3.3	pre-processing data	13
3.3.4	Frontend	14
3.3.5	Backend	14
3.3.6	Various plots	14
3.3.7	Implementing a Neural Network	17
3.3.8	Implementing a BNN	17
3.4	Procedure, Techniques and Methodologies,etc	18
3.4.1	DIFFERENT VERSIONS OF BNN	18
3.4.2	Models developed	19
3.4.3	STREAMLIT BASED WEB APP FOR EXPERT SYSTEM	20
4	Results and Discussions	22
4.1	BNN Models	22
4.1.1	BNN - VI Models	22
4.1.2	BNN - MC via Dropout	24
4.1.3	Early Stop	26
4.2	Web Application	28
4.2.1	SCREENSHOT - 1,2,3: APP INFORMATION	28
4.2.2	SCREENSHOT - 4,5: RUN THE APP	29
4.2.3	SCREENSHOT - 6: SEE THE SOURCE CODE	31
4.2.4	SCREENSHOT - 7: EXPERT SYSTEM	32
4.2.5	SCREENSHOT - 8: MARKDOWN REPORT	34
4.3	Comparison of Models	35
5	SUMMARY AND CONCLUSION	37

LIST OF FIGURES

3.1	Histograms of All Attributes.	15
3.2	Heatmap representing Correlation Matrix	15
3.3	Scatter Plots	16
3.4	Box Plots	16
4.1	MODEL V1 - BNN	22
4.2	MODEL V2 - BNN	23
4.3	MODEL Normal-1 BNN	23
4.4	MODEL Normal-2 BNN - BNN	23
4.5	Model Normal - BNN 1 and 2 for Comparison with Dropout	24
4.6	Model Drop-1 Accuracy and Loss	24
4.7	Model Drop-2 Accuracy and Loss	25
4.8	Model Drop-3 Accuracy and Loss	25
4.9	Model Normal - BNN 1 and 2 for Comparison with Early Stop	26
4.10	Model Early-1 Accuracy and Loss	26
4.11	Model Early-2 Accuracy and Loss	27
4.12	Model Early-3 Accuracy and Loss	27
4.13	Basic Home Page of the application	28
4.14	Basic Instructions and to-do of the app.	29
4.15	Full-Screen view of the Home Page.	29
4.16	Prediction form page	30
4.17	Prediction form page results	30
4.18	Code of the application: stream_app.py	31
4.19	Code of the application: stream_app.py	31
4.20	Expert System displaying steps.	32
4.21	Expert System addressing a user query -1	32
4.22	Expert System addressing a user query -2	33
4.23	Expert System addressing a user query -3	33

4.24	Markdown report - 1	34
4.25	Markdown report - 2	34
4.26	Comparison of all BNN Models	35
4.27	Comparison of all Dropout models with BNN Models	35
4.28	Comparison of all Early stop models with BNN Models	36

LIST OF TABLES

3.1	Standard Neural Network Model Summary	17
3.2	Bayesian Neural Network Model Summary	18

ABBREVIATIONS

ANN	Artificial Neural Networks
BNN	Bayesian Neural Networks
ES	Expert System
IIT	Indian Institute of Technology, Tirupati
M.Tech	Masters in Technology

NOTATION

$\mathbf{p}(\theta)$ probability distribution

CHAPTER 1

INTRODUCTION

The Expert System is a very old concept in the field of AI. Many expert systems were built for various domains and applications ranging from the first developed MYCIN Expert system for medical diagnosis to different machinery repair expert systems, credit card fraud detection systems, bug reporting systems, etc. Even though the applications are different, the underlying concept is the same. Every Expert system has the following components - Knowledge base, Inference Engine, and User(User Interface). In this project, we are developing an expert system for predictive maintenance.

For any kind of hardware/ machine, to have a longer lifetime and efficient usage, following best practices which improve its life cycle and maintenance of it through regular checking of it by an expert related to that machine is essential. This kind of periodic maintenance of a machine is costly for any entity or a person owning it. So here, predictive maintenance comes in handy. It can be achieved by the adoption of machine learning, automation or deep learning practices in its implementation. It helps in the reduction of maintenance costs by monitoring the health of machines.

Predictive Maintenance: It can be defined as condition-based maintenance where scheduling of maintenance takes place before the breakdown of a machine but only when certain conditions are satisfied or met. It is efficient and cost-effective when compared to periodically scheduled maintenance.

1.1 Introduction

1.1.1 Expert System

Definition: Expert system can be defined as a knowledge-based system where it acquires knowledge about a particular(specific and narrow) domain with the help of an expert from that domain and uses its inference engine(here it's BNN and not rule-based ones) to

address user queries. The user queries the expert system with the user interface provided to the user. It takes this input, processes it with the help of its domain knowledge and its inference engine, and results in the answer/output to the query.

The advantage of this expert system is that the knowledge base can be updated as per need to make the expert system with up-to-date knowledge. This system simulates human(expert) intelligence. It assists an expert in making better decisions but it is not a replacement. The advantage is that these systems are consistent in their predictions with better accuracy and are fast compared to humans.

The main disadvantage of these systems is the Knowledge acquisitions part which is a tedious task. We need to acquire the knowledge from the expert belonging to that domain.

1.1.2 Neural Networks

Neural Networks: These are the basic building blocks of the Deep learning domain. When compared to traditional learning approaches, we don't need to tell how to learn the data to the neural network. It automatically learns from the given observation(training) data, extracts the patterns, and applies the new data for predictions. One of the main disadvantages is that these are overconfident in their predictions which leads to uncertainty. So to overcome this and to have more dynamic predictions we are using Bayesian Neural networks where we use Bayesian inference in neural networks which gives us confident predictions and with certainty.

1.1.3 Bayesian Neural Networks(BNN)

BNN:The authors of the paper [Jospin *et al.* \(2020\)](#) "Hands-on Bayesian Neural Networks - a Tutorial for Deep Learning Users" defined it as a stochastic ANN trained using Bayesian inference. In simple terms, we use the probability of weights and biases and other parameters instead of singular values to the above terms. In the same paper as above, Stochastic ANN's are defined as ANN's in which stochastic components are used to simulate multiple possible model parameters along with their probability distributions. This can be considered as a equivalent to Ensemble learning(multiple different models

combined to have the most accurate predictions).

1.2 ES AND BNN

1.2.1 BAYESIAN NEURAL NETWORKS

Introduction

The Bayesian paradigm which is the base for Bayesian neural networks work on two basic and simple ideas,

- Probability of an event is defined as the belief in the occurrence of that event.
- Prior beliefs(probabilities) influence posterior beliefs(probabilities). This is best explained by the Bayes theorem.

Bayesian Neural Network: BNN = Stochastic Neural Network + Bayesian Inference.

Bayesian Neural Network is popularly defined as combination of Stochastic Neural Network and Bayesian Inference.

Implementation

Stochastic Neural Network = ANN + stochastic components(using probability distribution $p(\theta)$).

Stochastic Neural Network is an Artificial Neural Network with stochastic components which is useful to simulate multiple models similar to the process of ensemble learning. BNN's are better at quantifying the most consistent uncertainty for their predictions when compared to ANN's. Without over fitting the data, they are also efficient for small datasets.

- They can be treated as equivalent to ensemble learning.
Ensemble learning: it's a technique where we use multiple models instead of a single one and the combination of them will work with much better accuracy.
- Stochastic models are implemented using probabilistic graphical models(PGM's). After this, the initial priors are set in the network using various techniques.
- Various Bayesian inference algorithms are MCMC(Markov chain Monte Carlo) and variational inference, Bayes by back-propagation, learning the priors method.
- In practice, it's hard to implement the exact algorithm for the implementation of BNN's, so it's a general practice to approximate it with various techniques.
some of them are:
 1. Bayes via Dropout

2. Bayes via Stochastic Gradient Descent
3. Bayesian inference on the last n layers of the network.

Conclusion

Even though the overall idea might be easy where we are using probability distributions of weights in the network, the actual algorithmic implementation is difficult.

1.2.2 EXPERT SYSTEM ARCHITECTURE

An Expert system is an intelligent system that is based on the knowledge base which uses an inference engine to solve problems. The knowledge base would be restricted to a specific and narrow domain and with the help of an inference engine, it solves problems or answers user queries related to that domain. It also has a heuristics engine which helps in better predictions. These machines help the domain expert in taking better decisions. These have a wide range of domain applications.

Expert System = knowledge base + inference engine + user interface(optional)

Knowledge Base

Knowledge base: It's the strength of the expert system as the expert system derives its power from the knowledge base.

Knowledge base = organized collection of facts + heuristics knowledge

Knowledge base is combination of 'organized collection of facts' and 'heuristics knowledge'. The process of building a knowledge base by getting the knowledge base from the domain expert along with various other sources is known as **knowledge engineering**. The way we arrange or organize the knowledge in the knowledge base is defined as **knowledge representation**.

Methods:

- **Frame-based systems:** In this, frames represent the attributes or properties of the objects and relationships among them.
- **Production rules:** the most commonly used method where we represent knowledge in the form of production rules. Similar to if-else conditional rules.

Inference Engine

Inference engine: It combines all the knowledge from the knowledge base it has and generates its predictions for the specified input provided with the help of the user interface provided to the user. Here, in this project, we are using Bayesian neural networks as the inference engine of this expert system to get the predictions with confidence and to avoid uncertainty when compared it with standard neural networks.

User Interface

User Interface: it's an optional one where we can use this to test the expert system by any user by providing them with the interface or the testing can be manually done by the developer by giving the inputs to the system.

Conclusion

The overall stages of an Expert system are:

1. Identifying the domain for the Expert system.
2. Designing the system and developing it.
3. Testing and refining it.
4. Deploying and maintaining the system.

The main limitations of Expert system are the domain expertise and knowledge acquisition.

1.2.3 ADVANTAGES

The main advantages of Expert systems using BNN's are that

- An Expert System with better performance and confident predictions.
- Useful for experts in decision making and applicable across various other domains.

1.2.4 LIMITATIONS

The main disadvantages of Expert systems using BNN's are that

- Domain expertise availability.
- Extract and up-gradation of knowledge from expert.

- The high complexity of Bayesian neural networks implementation in practice increases the cost of the project to overcome the uncertainties

1.3 Problem Statement

We are developing an Expert system for the domain of Predictive Maintenance by using Bayesian Neural Networks as a part of its inference engine. Why use Bayesian Neural Networks when we can do it with Standard Neural Networks (General ones), It is because normal Neural Networks fit over the given data and becomes overconfident about their predictions. This causes them to be uncertain. To overcome this, we are implementing Bayesian Neural Networks for developing an inference engine for our expert system. These BNN's predicts with certainty and are dynamic and confident over their predictions. They achieve it by using probabilities for weights and biases instead of only integers.

1.4 Our Contribution

We have developed a web application for demonstrating the expert system using streamlit framework. It has 5 web pages in it and a drop-down menu to navigate to those pages within the app. The application contains all the models developed in the google co-lab notebook. These models are loaded into the app to get predictions.

The 5 web pages are:

- **Basic information page:** This feature in the application will enable the user to know more information the application.
- **Run the app page:** This feature in the application will enable the user to the application for getting predictions with the desired model.
- **See the source code page:** This feature in the application will enable the user to see the source code of the complete web application.
- **Expert system page:** This feature in the application will enable the user to run the expert system by answering the given questions.
- **Markdown report page:** This feature in the application will enable the user to see the report of the application written in markdown.

1.5 Organization of The Report

This thesis has been arranged in the following order.

- **Chapter 2: Background Information:** In this chapter, we discuss the existing works in the domain of Expert Systems and Bayesian Neural Networks.
- **Chapter 3: Materials and Methods:** In this chapter, we discuss the tools, technologies, procedures, techniques, and methodologies used in implementing this project.
- **Chapter 4: Results and Discussions:** In this chapter, we discuss the results that are obtained in this project during and after its implementation.
- **Chapter 5: Summary and Conclusion:** In this chapter, we discuss the summary of this thesis or report and the conclusion of this project.

CHAPTER 2

BACKGROUND INFORMATION

In this chapter, we discuss the existing works in the domain of Expert systems and Bayesian Neural Networks.

2.1 Expert System

The papers discussed for the literature review are :

[Adekunle *et al.* \(2018\)](#) "An Expert System for Automobile Repairs and Maintenance"

In this, the authors have designed an expert system that can detect various automobile problems instead of using a human expert for the same requirement. It's a complete GUI based expert system where a user can check the possible faults with causes and solutions to them and also diagnose the car with possible symptoms, causes and solutions to those problems.

[Kontargyri *et al.* \(2007\)](#) "An Expert System for Fault Diagnosis, Repairing and Maintenance of Electrical Machines"

In this, the authors have developed an expert system software tool that uses an SQL database with different access levels and with the use of suitable weights for the questions to define their priority. For this, a simple user interface is developed which can be used by all to effectively monitor the condition of electrical machines.

[Abd-Elhamid *et al.* \(2011\)](#) "Development of an Expert System for Maintenance and Repair of Masonry Barrages"

In this, the authors focus on building an expert for the domain which has very limited experts so as to carry forward the knowledge to others and also in better maintenance and early monitoring of barrages for repair to improve their life. In this, the expert system consists of only a knowledge base and inference engine.

[Simeón *et al.* \(2010\)](#) "An Expert System for fault Diagnostics in Condition-based Maintenance"

In this, the authors discussed condition-based maintenance which helps in monitoring the health of the machine and its various critical components, monitors its performance and help in the early repair of the machine which reduces a great amount of repair cost, logistic cost etc. Data Acquisition, signal processing, condition monitoring, health assessment, prognostics and decision support were the stages involved in this expert system.

[Çınar *et al.* \(2020\)](#) "Machine Learning in Predictive Maintenance towards Sustainable Smart Manufacturing in Industry 4.0"

In this, the authors have discussed Predictive Maintenance and its importance in the present industry 4.0. Various Machine learning techniques have been discussed in this paper which can help in fault detection and diagnosis which helps to have fewer failures and more usage of the machine.

[Akinluli *et al.* \(2015\)](#) "Development of an expert system for the repair and maintenance of bulldozer's work equipment failure"

In this, the authors have developed an expert system for the repair and Maintenance of bulldozer's equipment, which gives the solutions to the various problems like low or high hydraulic pressure, abnormal noise etc and helps in getting repaired quickly improves its performance and its life-cycle.

2.2 BNN

The papers discussed for the literature review are :

[Jospin *et al.* \(2020\)](#) "Hands-on Bayesian Neural Networks - a Tutorial for Deep Learning Users"

In this, the authors describe quantifying the uncertainty of deep learning models with the help of Bayesian inference and how it can be implemented with the help of Bayesian neural networks. They have described in detail all the steps involved in constructing the Bayesian neural networks and their practical implementations.

[Bykov et al. \(2021\)](#) "Explaining Bayesian Neural Networks"

In this, the authors discussed the limited transparency of BNN's and two perspectives of transparency about BNN's and their proposed way of explaining BNN's will help to get the most effective and insightful explanations. They have used various evaluation procedures and experimented with different datasets.

[Mourdoukoutas et al. \(2021\)](#) "A Bayesian Approach to Invariant Deep Neural Networks"

In this, the authors describe learning invariances directly from data with the help of posteriors distributions over various weight sharing schemes. and is outperforming non-invariant architectures. They have experimented with the different forms of the data.

[Rakesh and Jain \(2021\)](#) "Efficacy of Bayesian Neural Networks in Active Learning"

In this, the authors discussed active learning and the efficiency of Bayesian neural networks for active learning. They found that BNN's are more efficient than ensemble learning methods in capturing the uncertainty. In this, they experimented and compared various other techniques along with BNN's and with the use of various other datasets.

[Hassen and Rish \(2021\)](#) "Approximate Bayesian Optimisation for Neural Networks"

In this, the authors discussed the option of exploring with BNN's instead of GP's(Gaussian Processes) for building the model distributions for given functions and how it improved the performance and efficiency of the system. They have experimented and bench-marked the effect of priors on BNN's with HPO Benchmark.

[Nguyen et al. \(2021\)](#) "Structured Dropout Variational Inference for Bayesian Neural Networks"

In this, the authors proposed a new technique called Variational Structured Dropout which is inspired by the Bayesian interpretation of the Dropout regularization technique. In this, the VSD technique introduces an adaptive regularization term with various desired properties which increases the model to make more generalized predictions. Besides this, they have discussed the scalability of their VSD technique and explicit regularization of the Variational Structured Dropout(VSD) technique.

[Chang \(2021\)](#) "Bayesian Neural Networks: Essentials"

In this, the author explained the Bayesian neural networks and how they are built and implemented with examples. They concluded that in practice, it's very inefficient to use due to its cost for complexity in implementing for all layers in the network, instead, using a variation of BNN mostly hybrid ones are more suitable in practice. The author described Bayesian inference using Markov Chain Monte Carlo(MCMC) inference and variational inference techniques and the use of Gaussian priors and posteriors besides bayesian priors and posteriors which are used in general.

CHAPTER 3

MATERIALS AND METHODS

3.1 About this Chapter

In this chapter, we describe the tools and technologies used, procedures, techniques and methodologies that are involved in this project.

3.2 Technologies used.

Streamlit: Streamlit [Streamlit](#) is one of the most popular open-source python libraries in the domain of machine learning and data science projects. It is useful in creating and sharing web applications in less time when compared with other frameworks. This framework/library is compatible with almost all the other popular python libraries such as sci-kit-learn, TensorFlow, Keras, etc. It uses markdown to write the textual content in the app which automatically gets converted into a full HTML web page for your application. It has various built-in features to include, forms, images, code etc in the web application.

TensorFlow Probability (TFP): is one of the python library [Dillon *et al.* \(2017\)](#) that is built on the most popular and powerful python library i.e TensorFlow which enables an easier way for combination of the probabilistic concepts and models with deep learning techniques which would be running on our modern hardware like TPU and GPU. It's mostly used by people who want to combine domain knowledge with deep learning to have a better understanding of data and to make better predictions. Some of those who use TFP are data scientists, statisticians, ML researchers.

Since TFP is built on TensorFlow, we can do all functionalities and steps like building a model, fitting the model onto the data and deploying a model by implementing it by using only one language throughout the life-cycle of the model.

TensorFlow: It is an open-source library [Abadi et al. \(2015\)](#) or a framework available in multiple languages and is used for machine learning and artificial intelligence applications. It focuses on deep learning networks.

3.3 Dataset, Front end, Back-end

3.3.1 Dataset

The dataset for this project is taken from UCI Dataset Repository [Dua and Graff \(2017\)](#) titled "AI4I 2020 Predictive Maintenance Dataset Data Set" provided by Stephan Matzka, School of Engineering - Technology and Life, Hochschule Wirtschaft Berlin, 12459 Berlin, Germany. It contains data of 10000 rows 1 for each data point and 14 columns 1 for each feature. The target variable 'machine failure' indicates whether the machine has failed for that particular machine point in the dataset only if any one of the independent failures has occurred for that machine.

3.3.2 Data Description

Unique values in each attribute of the dataset:

UDI: 10000, Product ID: 10000, Type: 3, Air temperature [K]: 93, Process temperature [K]: 82, Rotational speed [rpm]: 941, Torque [Nm]: 577, Tool wear [min]: 246, Machine failure: 2, TWF: 2, HDF: 2, PWF: 2, OSF: 2, RNF: 2.

datatype: int64

3.3.3 pre-processing data

Pre-processing the data:

Ordinal Encoding:

```
df['Type'].unique() : array(['M', 'L', 'H'], dtype=object)
```

- L, M, H are three types representing low (50% of all products), medium (30%), and high (20%) as product quality variants respectively.

- Converting this categorical data to numerical with class 0, 1, 2 for L, M, H respectively using OrdinalEncoder from the sklearn library.
- One-hot encoding is not suitable for ordinal data so we use an Ordinal encoder.
- This gives categories converted into integers.
- This sorts all the categories present and assigns values to them in alphabetical order.
0 for H
1 for L
2 for M

Getting all numerical data: 'Type', 'Air temperature [K]', 'Process temperature [K]', 'Rotational speed [rpm]', 'Torque [Nm]', 'Tool wear [min]', 'TWF', 'HDF', 'PWF', 'OSF', 'RNF', 'Machine failure'

Creating training and evaluation datasets : Using the *train_test_split* function, I split the data into training and testing in the ratio of 70% - 30%.

3.3.4 Frontend

Frontend for this web application is done with the help of streamlit which actually takes care of the backend model to be loaded into the app. It has all built-in options to easily build a web app that too in python language and a mix of markdown for displaying and formatting the content.

3.3.5 Backend

Backend part for this web application is working on the model after taking the input from the user and it is handled both by the streamlit and other python libraries like TensorFlow for loading the model and sending these variable values to it to get its predictions.

3.3.6 Various plots

All these are the plots that are plotted during the implementation of the project.

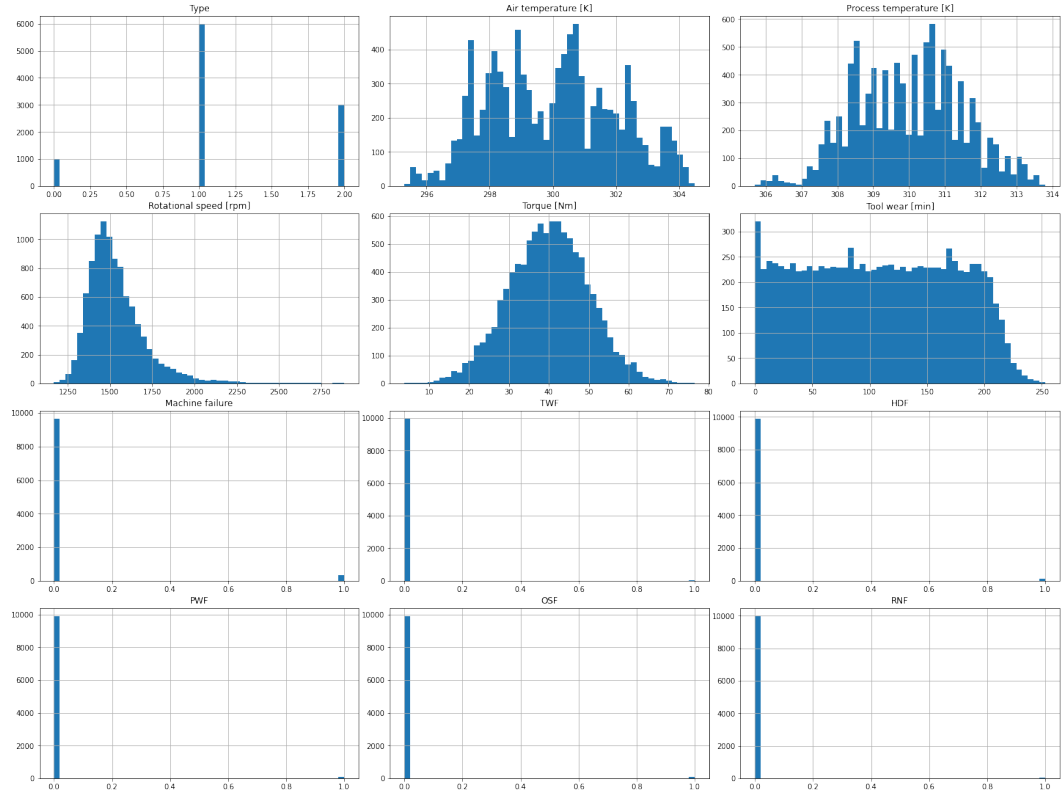


Figure 3.1: Histograms of All Attributes.

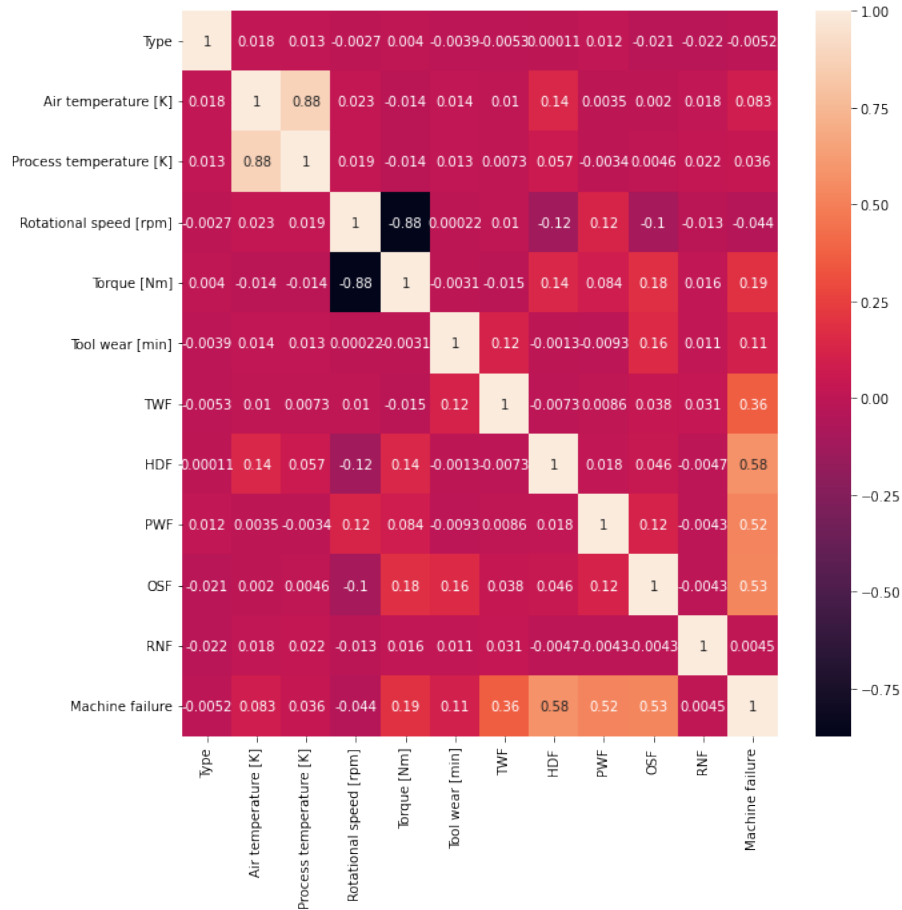
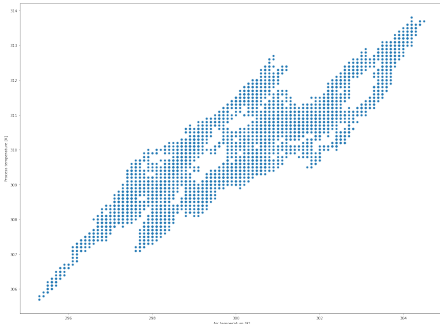
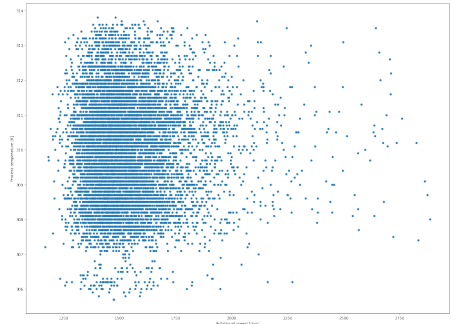


Figure 3.2: Heatmap representing Correlation Matrix

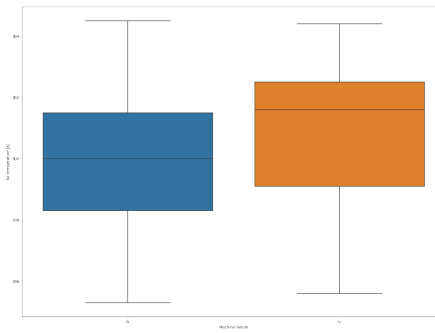


(a) Scatter Plot of process and Air temperatures

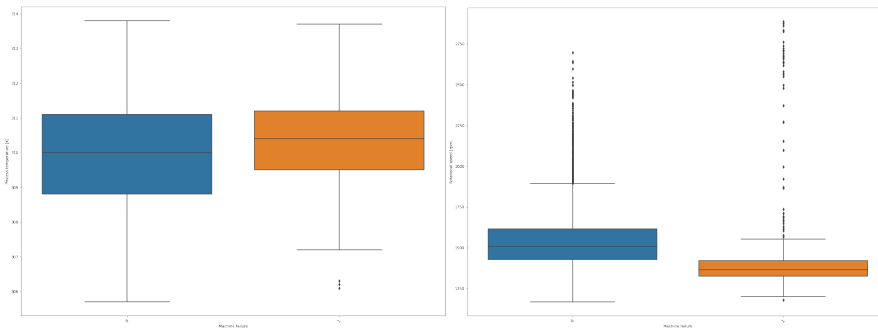


(b) Scatter Plot of process and Rotational Speed

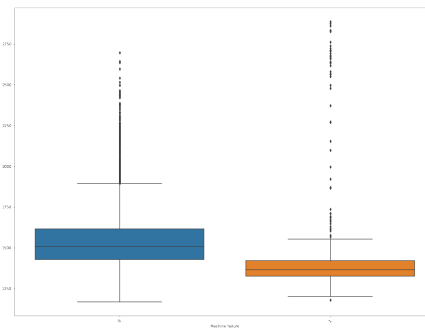
Figure 3.3: Scatter Plots



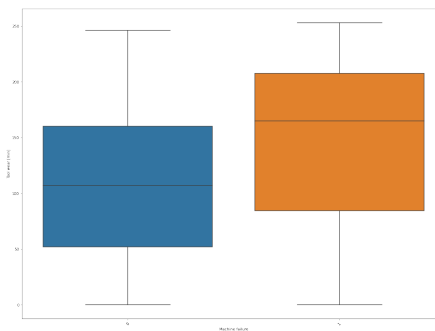
(a) Box plot of Air temperature and Machine Failure.



(b) Box plot of Process temperature and Machine Failure.



(c) Box plot of Rotational Speed and Machine Failure.



(d) Box plot of Tool wear and Machine Failure.

Figure 3.4: Box Plots

3.3.7 Implementing a Neural Network

Standard Neural Network:

No of epochs: 50

Learning rate: 0.001

Test : Accuracy: 0.9690

Test loss: 0.1385

Layers: Dense size = 5 ; parameters = 65

Dense size = 3 ; parameters =18

Dense size = 1 ; parameters =4

train accuracy: 0.9649, loss: 0.1522 after 50 epochs

test accuracy: 0.9690, loss: 0.1385

Model: "sequential"

Layer (type)	Output Shape	Param
dense (Dense)	(None, 5)	65
dense_1 (Dense)	(None, 3)	18
dense_2 (Dense)	(None, 1)	4

Table 3.1: Standard Neural Network Model Summary

3.3.8 Implementing a BNN

Bayesian Neural Network (BNN):

No of epochs: 50

Learning rate: 0.001

Test : Accuracy: 0.9686

Test loss: 0.450

Test accuracy: 0.9686 after 50 epochs and test loss: 0.450

Model: "sequential_2"

Layer (type)	Output Shape	Param
dense_flipout (DenseFlipout)	(None, 16)	368
dense_flipout_1 (DenseFlipout)	(None, 6)	198
dense_flipout_2 (DenseFlipout)	(None, 3)	39

Table 3.2: Bayesian Neural Network Model Summary

3.4 Procedure, Techniques and Methodologies,etc

3.4.1 DIFFERENT VERSIONS OF BNN

These different versions of Bayesian Neural Networks are developed by using these below-mentioned techniques.

VI (VARIATIONAL INFERENCE)

Variational Inference is also known as a posterior distribution over the latent variables and it can be defined as the conditional distribution over the latent variables given the observations and parameters.

MC VIA DROPOUT

Monte Carlo Dropout is an improvement of the regular dropout (in which different neurons are dropped) technique which can be interpreted as a Bayesian approximation of a probabilistic model in which we consider many different neural networks as Monte-Carlo samples out of all models which are available. In most cases, it improves the performance of the model.

EARLY STOP

It can be defined as stopping the training of a neural network when a monitored value has not improved for a certain number of iterations. This usually improves the performance

of the model as the number of epochs run for a model is usually less when compared to it without an early stopping technique-based neural network.

patience: It is the limit on the number of epochs with no improvement. After reaching this count or limit, the training of the model will be stopped irrespective of the total epochs mentioned.

monitor: It describes the quantity(here it is loss which is being monitored.) which is to be monitored during the training of the model.

3.4.2 Models developed

The following were the steps taken in implementing all kinds of (Bayesian Neural Networks) BNNs.

- Defining a model.
- Fitting the model onto the data.
- Evaluating the model.
- Plotting different parameters of the model for comparison.
- Saving the model as a file for including it in the web application.
- Saving the model architecture as an image.

All these BNN models with different versions in them are described below. **Variational Inference :**

1. Normal BNN -1: 3 - layers, epochs = 25, learning rate = 1e-06
2. Normal BNN -2: 4 - layers, epochs = 15, learning rate = 1e-06
3. Normal BNN -1: 3 - layers, epochs = 40, learning rate = 1e-06
4. Normal BNN -2: 4 - layers, epochs = 40, learning rate = 1e-06
5. Normal BNN -1: 3 - layers, epochs = 50, learning rate = 1e-06
6. Normal BNN -2: 4 - layers, epochs = 50, learning rate = 1e-06
7. Model tfp v1: 3 - layers, epochs = 40, learning rate = 0.002
8. Model tfp v2: 3 - layers, epochs = 80, learning rate = 0.005

Monte Carlo - Dropout: learning rate = 1e-06, epochs=40

1. Dropout 1 : dropout value = 0.2
2. Dropout 2 : dropout value = 0.35
3. Dropout 3 : dropout value = 0.5

Early Stopping: learning rate = 0.005, epochs=50

1. Early Stop 1: patience = 3
2. Early Stop 2: patience = 4
3. Early Stop 3: patience = 2

3.4.3 STREAMLIT BASED WEB APP FOR EXPERT SYSTEM

Contains information about the app.

The main homepage of the web app displays basic info and a logo of the application along with a sidebar drop-down menu containing all the options to move to various pages or sections in the application.

Contains an option to load any desired model from the above notebook file.

- It has a default model loaded into the application initially but has the drop-down menu to select any model for getting the predictions as per the requirement.
- After building a model in the colab file, saving the model in either hd5 format or as a folder using `save_model()` function, and downloading them to the local folder into the streamlit web app folder so that it can be loaded. This model is loaded into the web app using the same technique of the `load_model()` function.

Contains an option for any user to enter values and get predictions from the model

- Has 11 input fields for all the variables to be entered by the user.
- This input is passed to the above-loaded model and the predicted value from the model is displayed in the app.
- As mentioned, It has a drop-down menu to select any model for getting the predictions as per the requirement before submitting the data to the application.

Expert System web page

The expert system page was completed by following the below steps.

Steps:

- Queries here can be answered to get the advice from Expert System.
- Users can select the appropriate option below each query to submit to the system.
- Based on the option selected, a certain value gets assigned to that variable.
- All these values will be sent to the model to get the prediction.
- The result value from the model is either 0 or 1.
- This value is converted into advice for the user for his specific query from the Expert System Model.
- From the model, if it is safe, just print that the machine is safe and if not, then based on the query value for the last 4 values for various machinery failures, print accordingly the cause of failure and a possible solution to it.

Here, in this Expert System, I have given a series of 4 questions which can be answered by the user by selecting different options available in the drop-down for each question,

based on the options selected, values are assigned to the related variables, and for other variables through different calculations. These numbers values which are assigned are sent to the default model which is loaded for prediction and the result value is again converted into answers which the user can understand and giving him problems and solutions based on the queries answered by user.

So to summarize, the app contains:

- Information about the app and its usage/code.
- Option to load any desired model on the prediction page of 11 fields.
- Use that selected model from the option to predict the output based on the given inputs.
- A page for normal users to select an option from a list of common queries and based on that query linked to the model gets the advice displayed on the screen.
- A markdown report page to summarize the results and conclusion including the technique implemented.

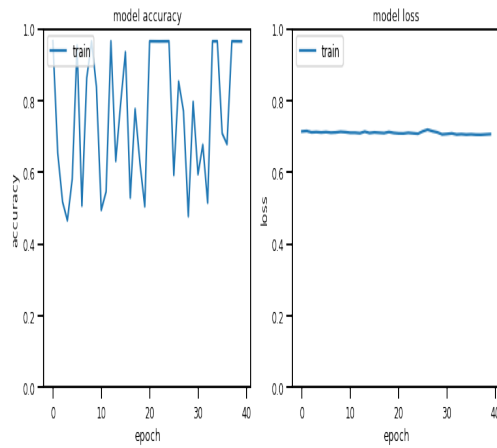
CHAPTER 4

Results and Discussions

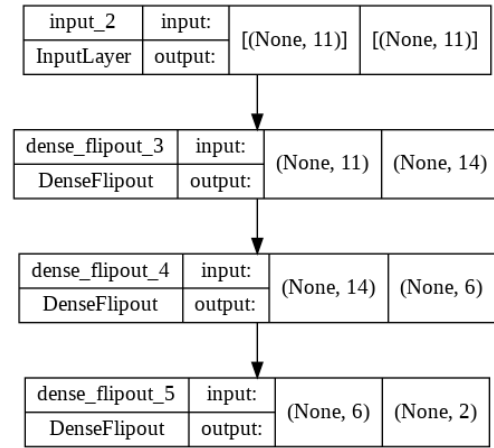
In this chapter, we discuss the results that are obtained in this project implementation.

4.1 BNN Models

4.1.1 BNN - VI Models

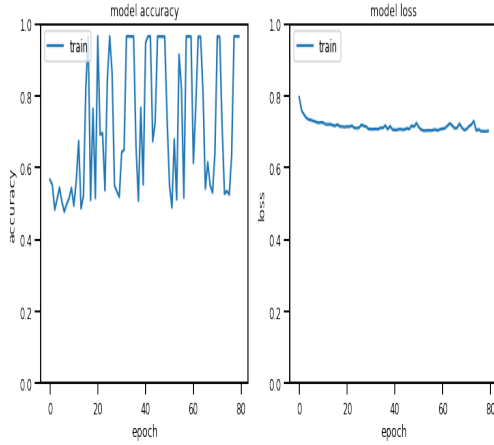


(a) Model V1 Accuracy and Loss

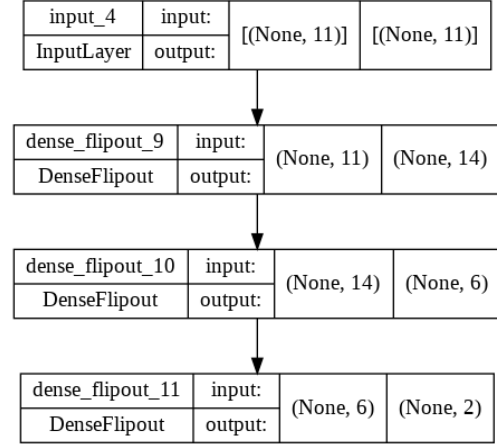


(b) Model V1 Architecture

Figure 4.1: MODEL V1 - BNN

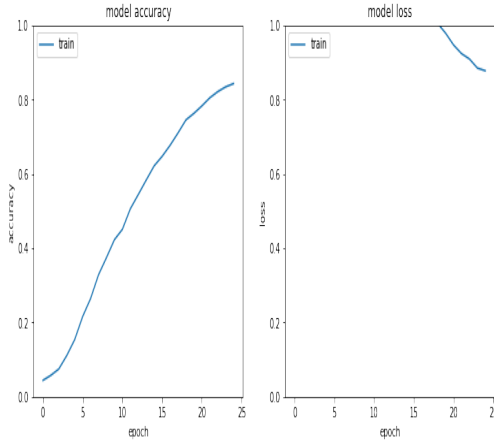


(a) Model V2 Accuracy and Loss

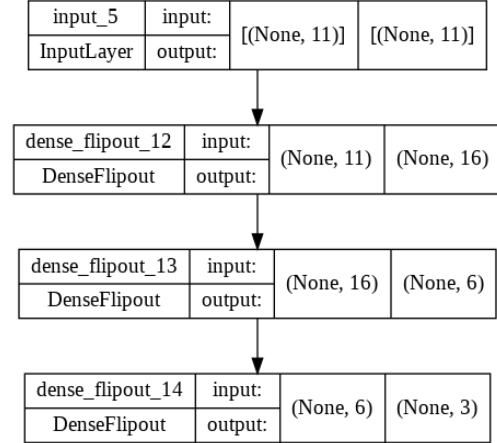


(b) Model V2 Architecture

Figure 4.2: MODEL V2 - BNN

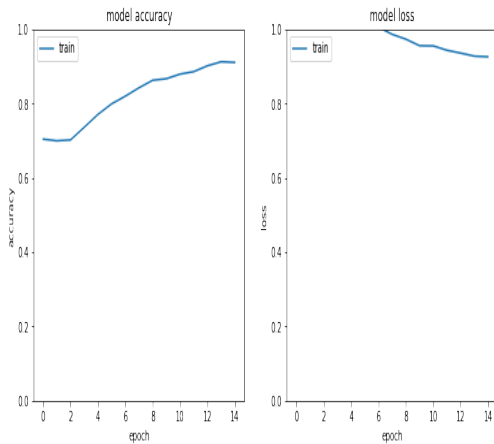


(a) Model Normal-1 BNN Accuracy and Loss

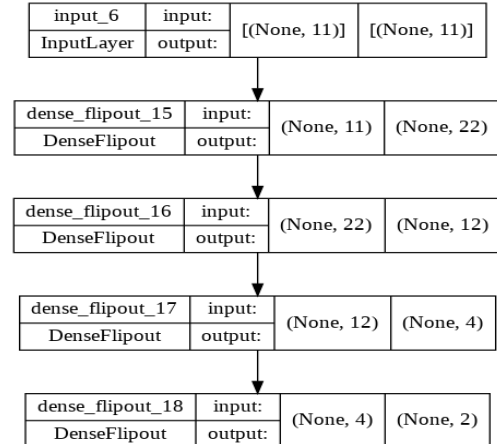


(b) Model Normal-1 BNN Architecture

Figure 4.3: MODEL Normal-1 BNN



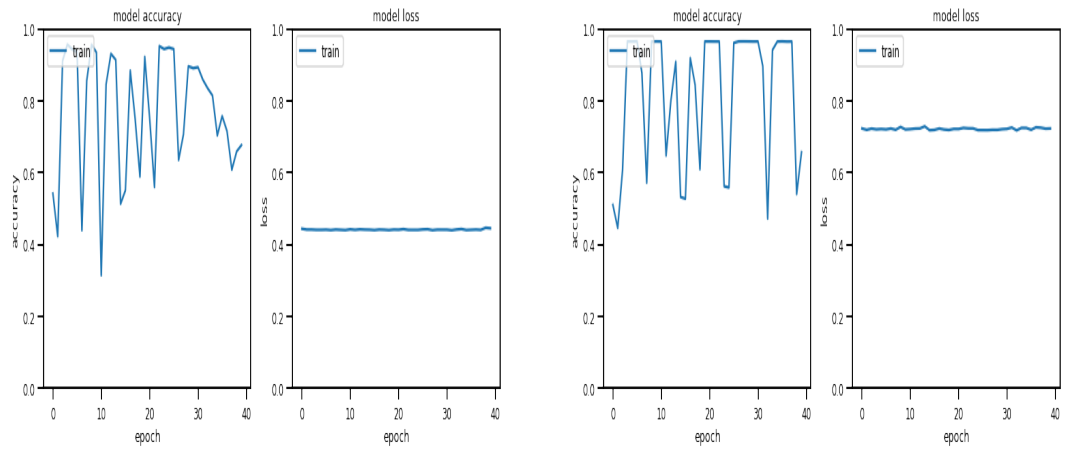
(a) Model Normal-2 BNN Accuracy and Loss



(b) Model Normal-2 BNN Architecture

Figure 4.4: MODEL Normal-2 BNN - BNN

4.1.2 BNN - MC via Dropout



(a) Model Normal-Cmp-1 Accuracy and Loss (b) Model Normal-Cmp-2 Accuracy and Loss

Figure 4.5: Model Normal - BNN 1 and 2 for Comparison with Dropout

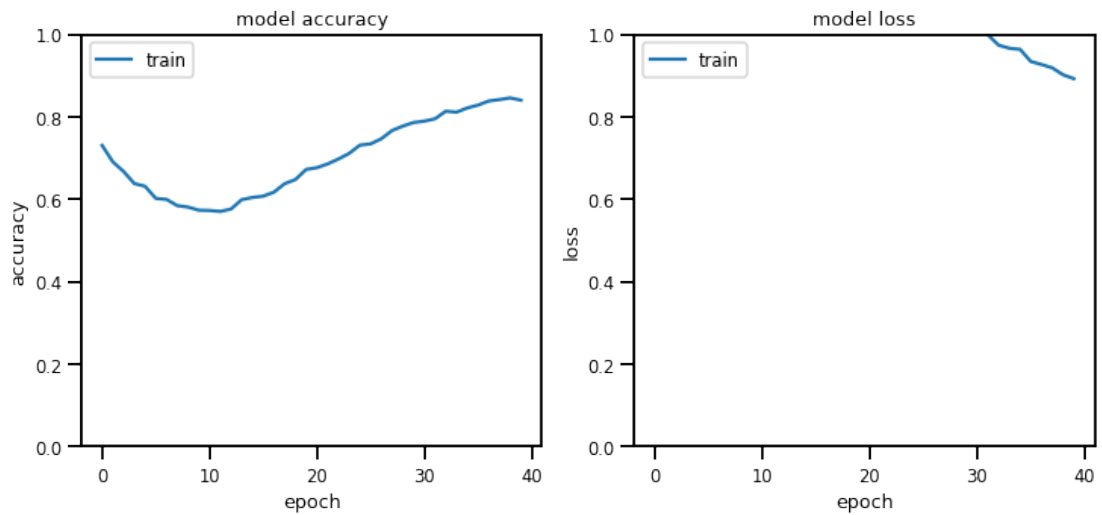


Figure 4.6: Model Drop-1 Accuracy and Loss

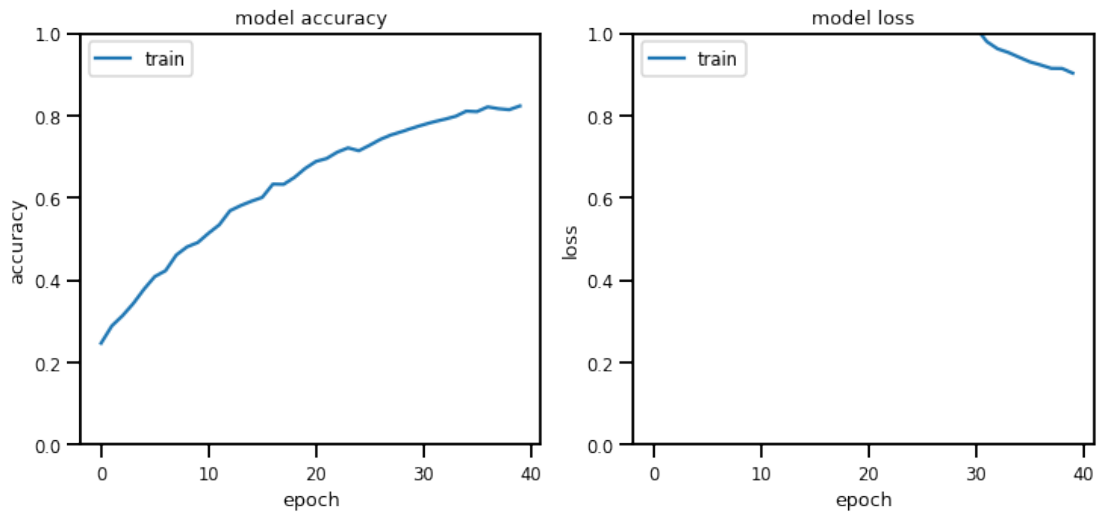


Figure 4.7: Model Drop-2 Accuracy and Loss

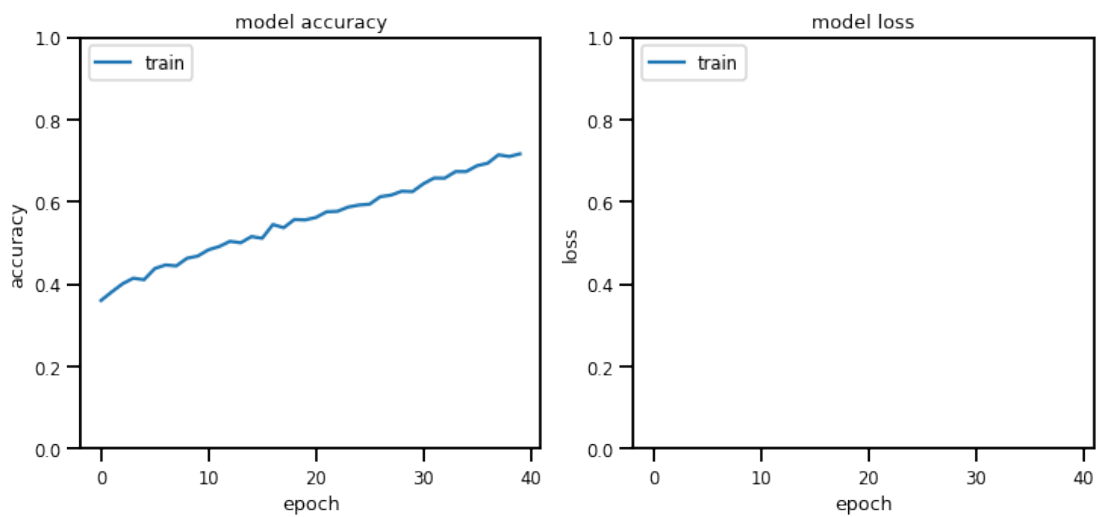
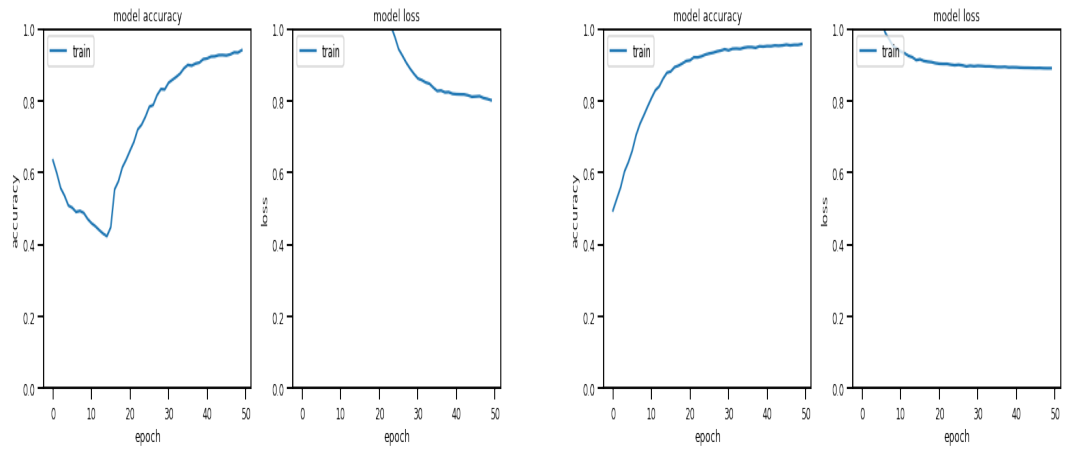


Figure 4.8: Model Drop-3 Accuracy and Loss

4.1.3 Early Stop



(a) Model Normal-Cmp-1 Accuracy and Loss (b) Model Normal-Cmp-2 Accuracy and Loss

Figure 4.9: Model Normal - BNN 1 and 2 for Comparison with Early Stop

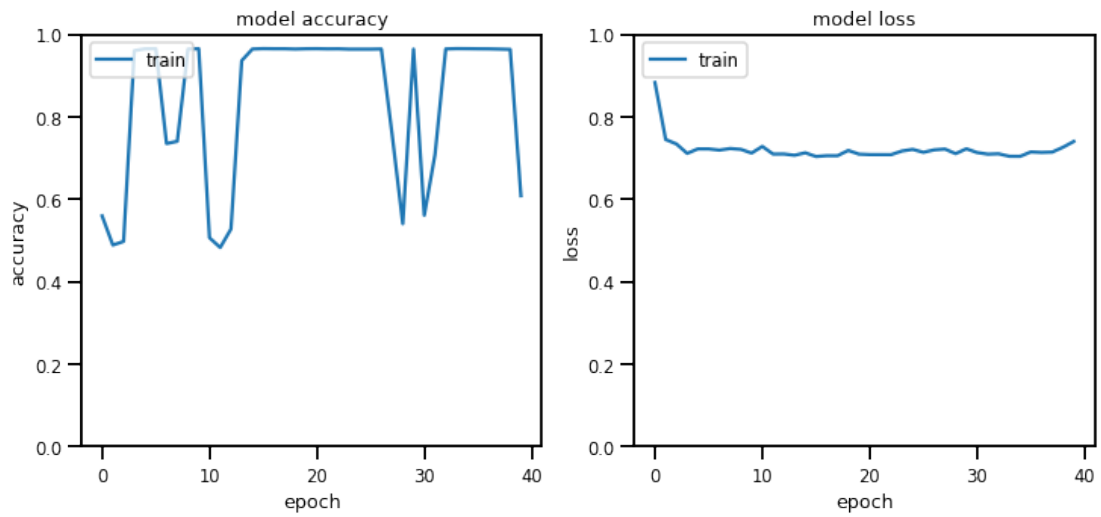


Figure 4.10: Model Early-1 Accuracy and Loss

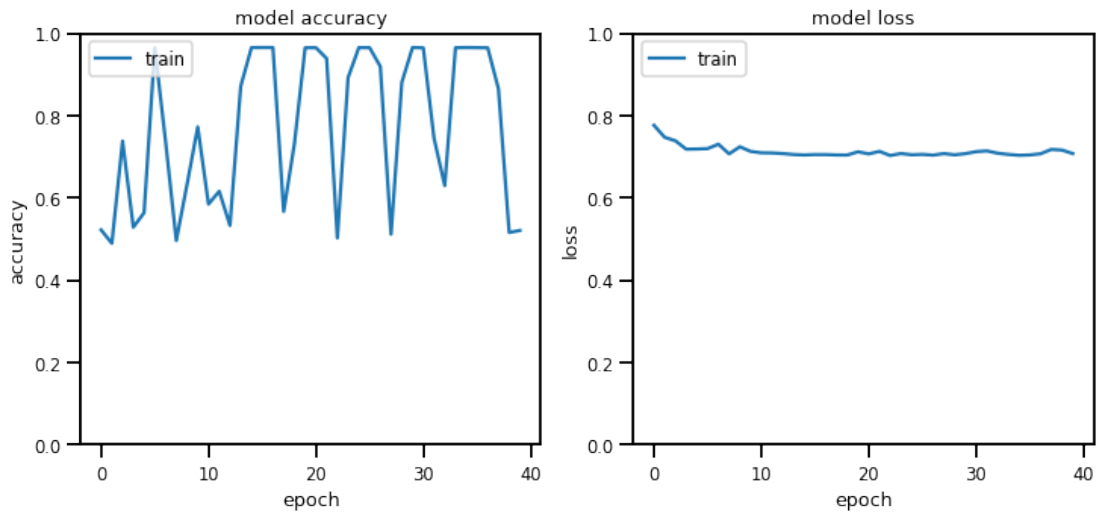


Figure 4.11: Model Early-2 Accuracy and Loss

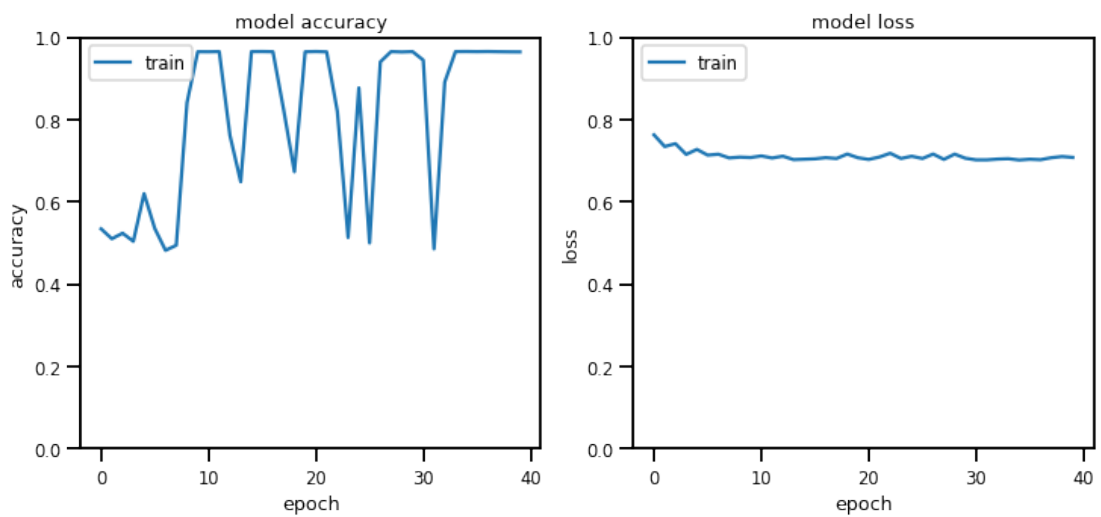


Figure 4.12: Model Early-3 Accuracy and Loss

4.2 Web Application

It has two main containers, one is a sidebar that has a drop-down menu to load or move into different pages or sections available in the application and the other one is the contents of the application.

4.2.1 SCREENSHOT - 1,2,3: APP INFORMATION



Figure 4.13: Basic Home Page of the application

This is the basic homepage of the web application. It has a logo for the web app and basic information related to the app. On the left side of the application, we have a drop-down menu with which we can navigate into various pages in the app.

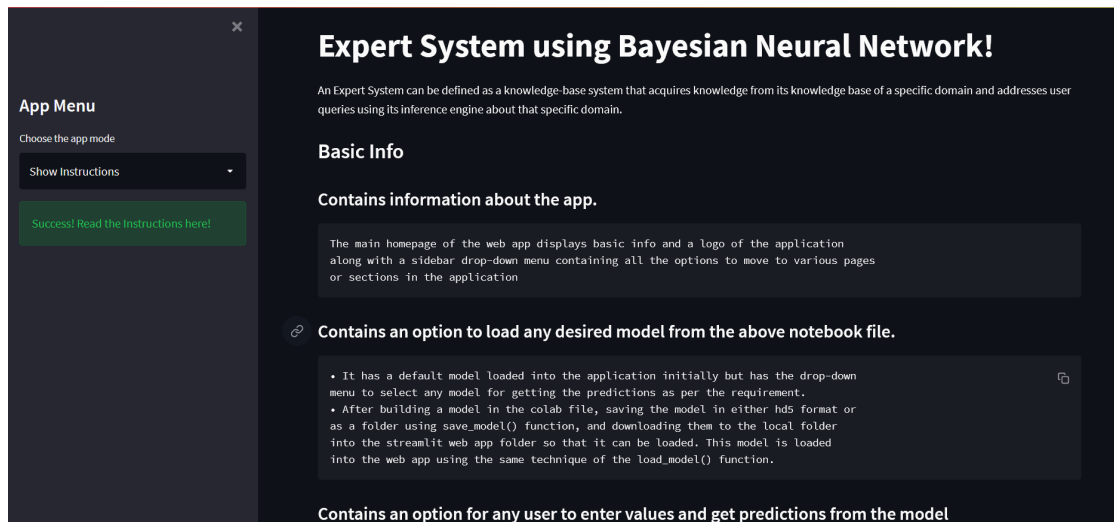


Figure 4.14: Basic Instructions and to-do of the app.
Displays the app's basic information and a checklist to track the progress of developing the application.

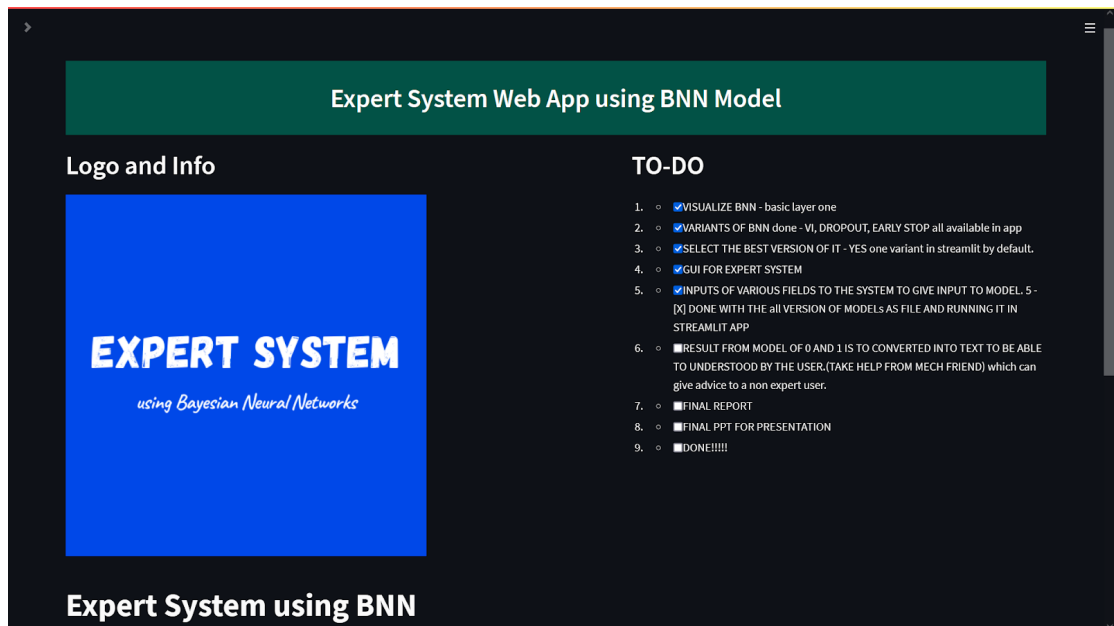


Figure 4.15: Full-Screen view of the Home Page.

4.2.2 SCREENSHOT - 4,5: RUN THE APP

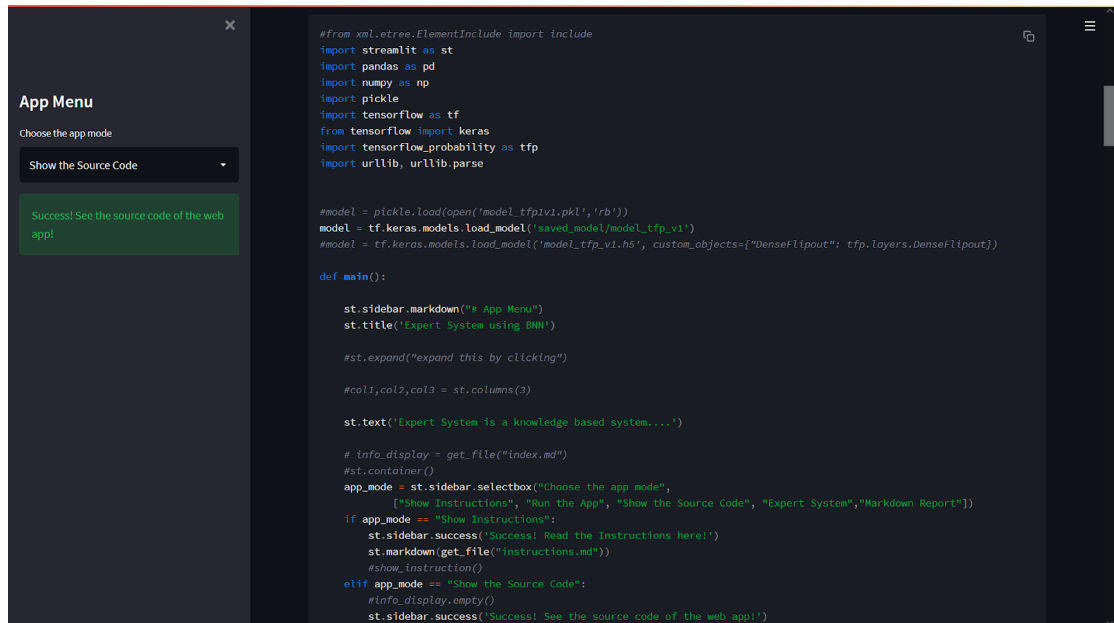
It displays the form where the user can enter the value into the app to get the predictions from the model.

Figure 4.16: Prediction form page
Form to be filled with all the variables and also option to select the model.

Figure 4.17: Prediction form page results
Result being displayed from the model for given inputs.

4.2.3 SCREENSHOT - 6: SEE THE SOURCE CODE

It has the option to view the source code of the application.



The screenshot shows a web application interface on the left and its source code on the right. The interface has a sidebar titled "App Menu" with a "Choose the app mode" dropdown and a "Show the Source Code" button. A green message box says "Success! See the source code of the web app!". The source code is a Python script using Streamlit, Keras, and TensorFlow to load a model and display information.

```
#from xml.etree.ElementInclude import include
import streamlit as st
import pandas as pd
import numpy as np
import pickle
import tensorflow as tf
from tensorflow import keras
import tensorflow_probability as tfp
import urllib, urllib.parse

#model = pickle.load(open('model_tfp_v1.pkl','rb'))
model = tf.keras.models.load_model('saved_model/model_tfp_v1')
#model = tf.keras.models.load_model('model_tfp_v1.h5', custom_objects={"DenseFlipout": tfp.layers.DenseFlipout})

def main():

    st.sidebar.markdown("# App Menu")
    st.title('Expert System using BNN')

    #st.expand("expand this by clicking")

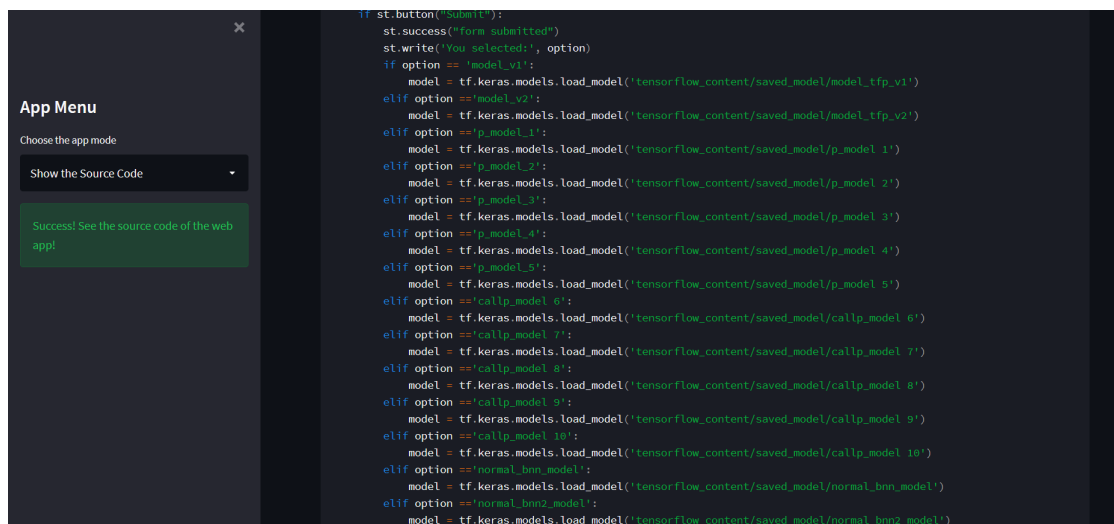
    #col1,col2,col3 = st.columns(3)

    st.text('Expert System is a knowledge based system....')

    # info_display = get_file("Index.md")
    #st.container()
    app_mode = st.sidebar.selectbox("Choose the app mode",
    ["Show Instructions", "Run the App", "Show the Source Code", "Expert System","Markdown Report"])

    if app_mode == "Show Instructions":
        st.sidebar.success('Success! Read the instructions here!')
        st.markdown(get_file("Instructions.md"))
        #show_instruction()
    elif app_mode == "Show the Source Code":
        #info_display.empty()
        st.sidebar.success('Success! See the source code of the web app!')
```

Figure 4.18: Code of the application: stream_app.py



The screenshot shows a web application interface on the left and its source code on the right. The interface has a sidebar titled "App Menu" with a "Choose the app mode" dropdown and a "Show the Source Code" button. A green message box says "Success! See the source code of the web app!". The source code is a Python script using Streamlit and Keras to load various models based on the selected app mode.

```
if st.button("Submit"):
    st.success("Form submitted")
    st.write('You selected:', option)
    if option == 'model_v1':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/model_tfp_v1')
    elif option == 'model_v2':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/model_tfp_v2')
    elif option == 'p_model_1':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/p_model 1')
    elif option == 'p_model_2':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/p_model 2')
    elif option == 'p_model_3':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/p_model 3')
    elif option == 'p_model_4':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/p_model 4')
    elif option == 'p_model_5':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/p_model 5')
    elif option == 'callp_model_6':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/callp_model 6')
    elif option == 'callp_model_7':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/callp_model 7')
    elif option == 'callp_model_8':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/callp_model 8')
    elif option == 'callp_model_9':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/callp_model 9')
    elif option == 'callp_model_10':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/callp_model 10')
    elif option == 'normal_bnn_model':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/normal_bnn_model')
    elif option == 'normal_bnn2_model':
        model = tf.keras.models.load_model('tensorflow_content/saved_model/normal_bnn2_model')
```

Figure 4.19: Code of the application: stream_app.py

4.2.4 SCREENSHOT - 7: EXPERT SYSTEM

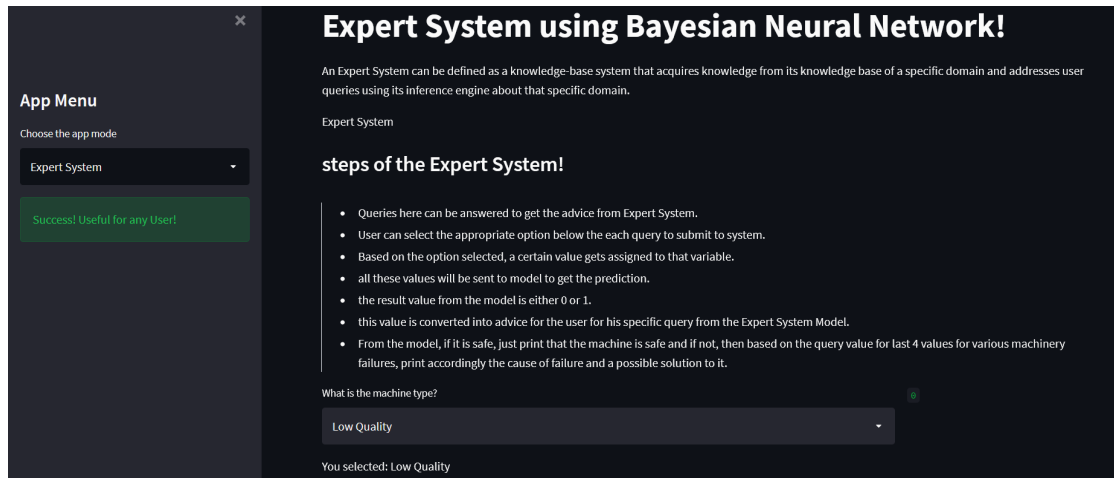


Figure 4.20: Expert System displaying steps.

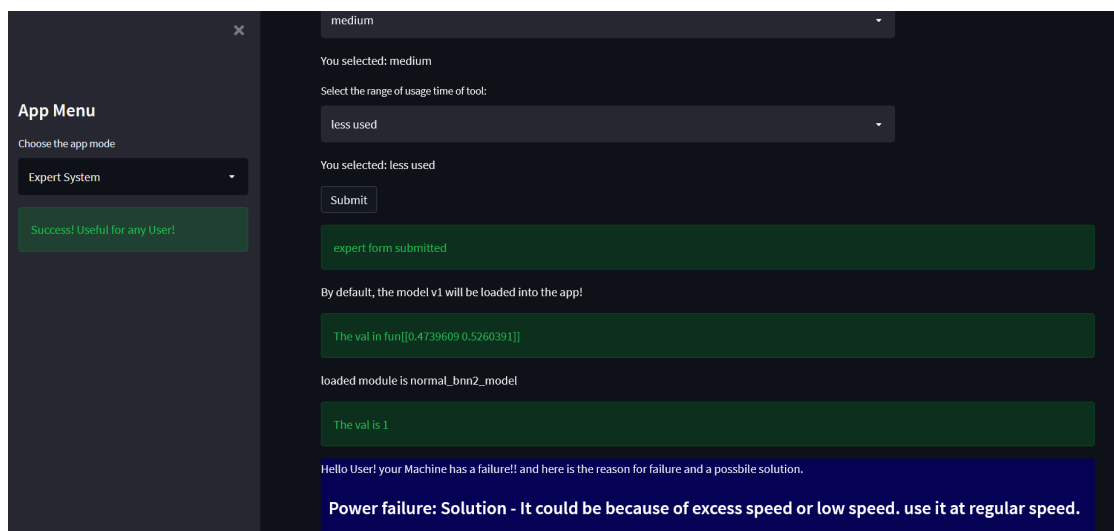


Figure 4.21: Expert System addressing a user query -1
Expert System addressing a user query by giving him/her a solution to the most probable problem caused.

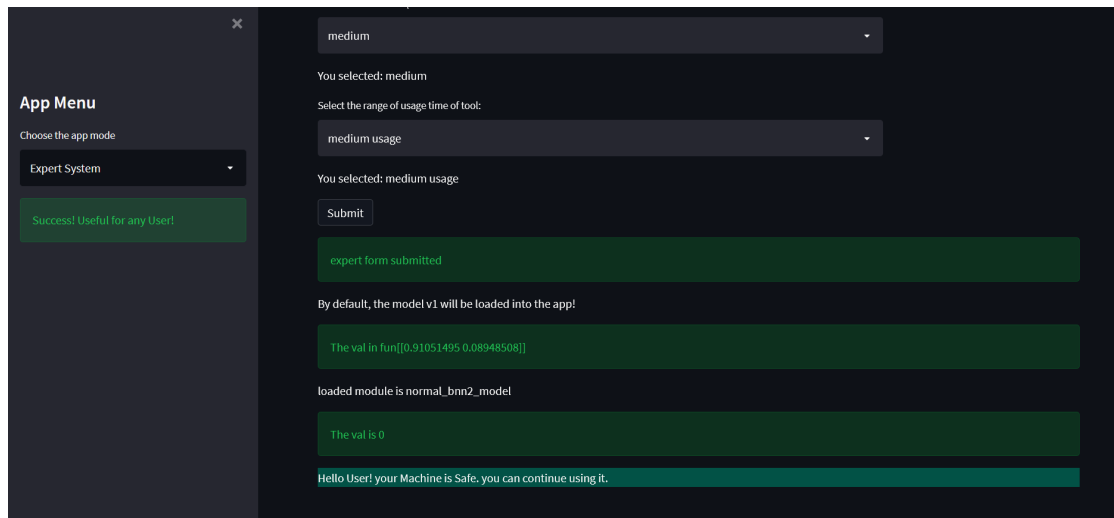


Figure 4.22: Expert System addressing a user query -2
Expert System addressing a user query by giving him/her answer that the machine is safe(good condition) it and can continue using it.

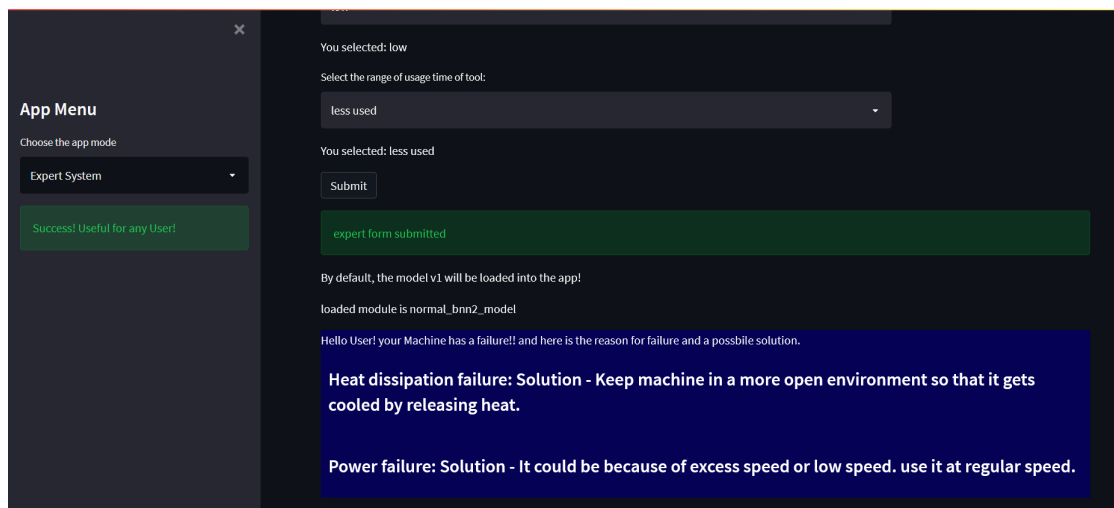


Figure 4.23: Expert System addressing a user query -3
Expert System addressing a user query by giving him/her solutions to the list of most probable problems caused.

4.2.5 SCREENSHOT - 8: MARKDOWN REPORT

It contains an option to display the markdown report content in the application.

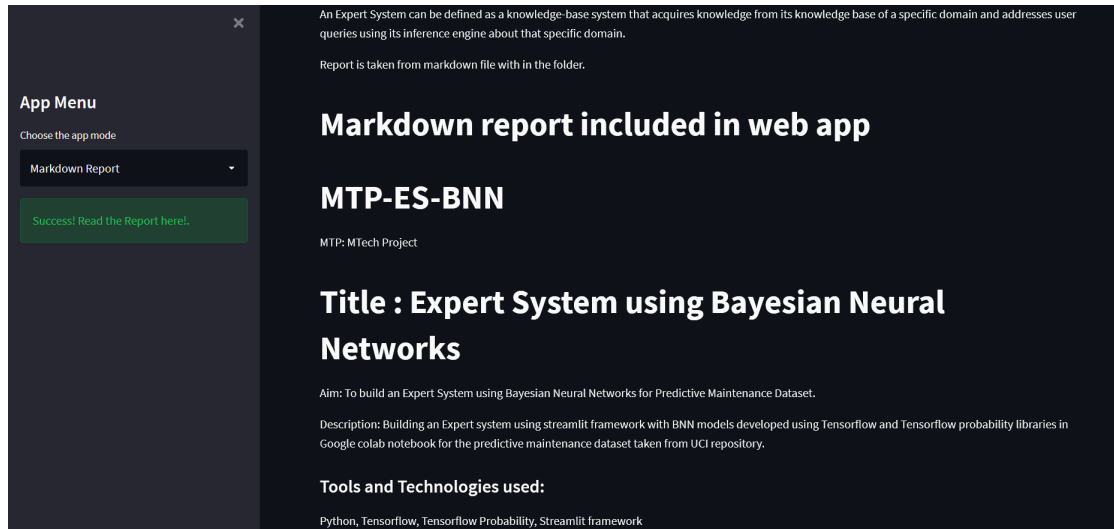


Figure 4.24: Markdown report - 1
Markdown report containing all the basic details from the report.

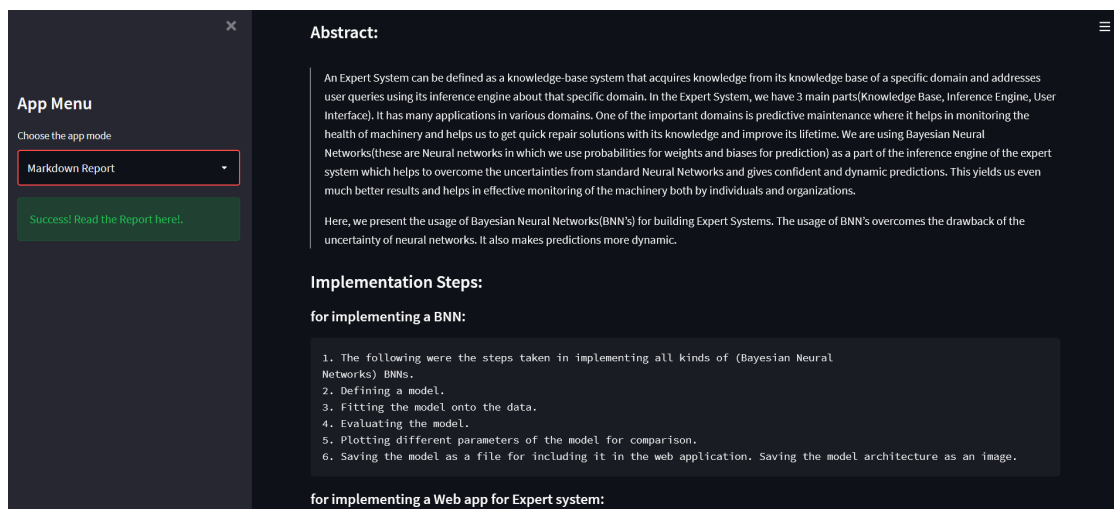


Figure 4.25: Markdown report - 2
Markdown report containing all the basic details from the report.

4.3 Comparison of Models

Below, are the plots for comparison of various model's performance using various metrics.

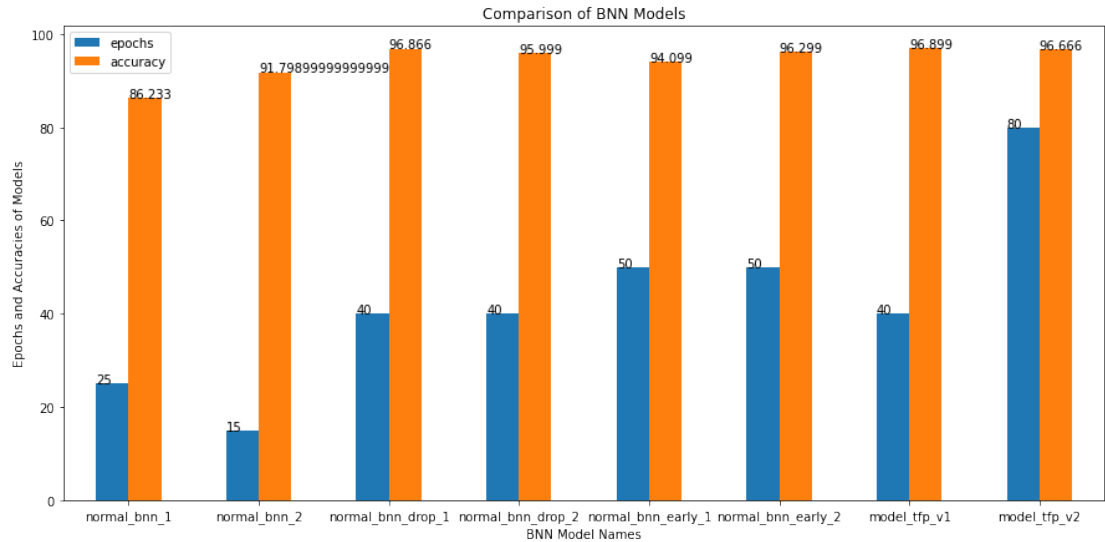


Figure 4.26: Comparison of all BNN Models
Comparison of the number of epochs and accuracy's for these models.

From the above comparison plot, we can observe that irrespective of their epochs and learning rates, all the models have almost all good accuracies ranging from 86% to 97% (approximately).

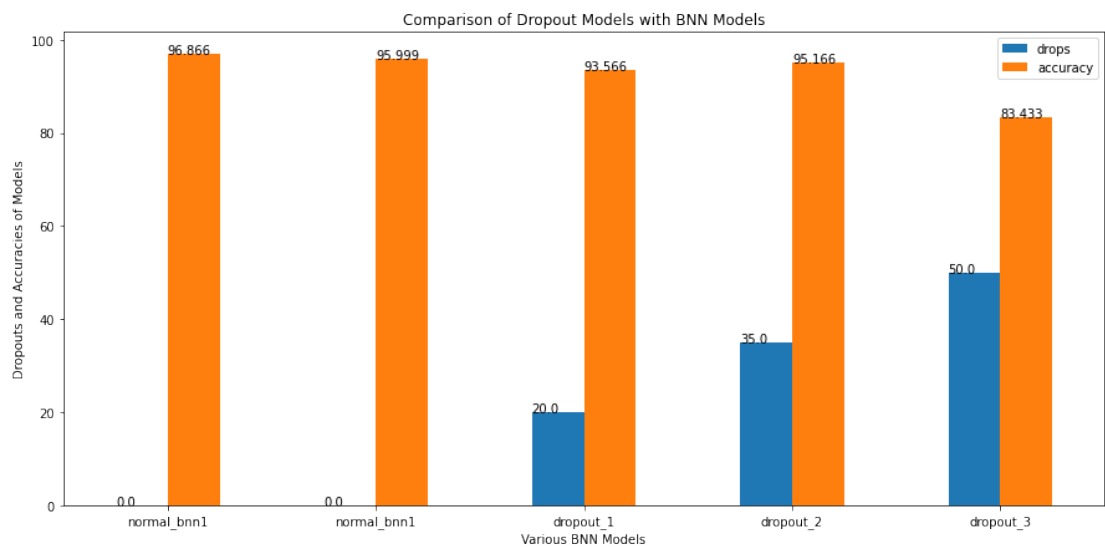


Figure 4.27: Comparison of all Dropout models with BNN Models
Comparison of the dropout percentage and accuracy's for these models.

In dropout BNN models, the model with dropout value as 0.35 (35%) is performing better when compared to other two dropout models.

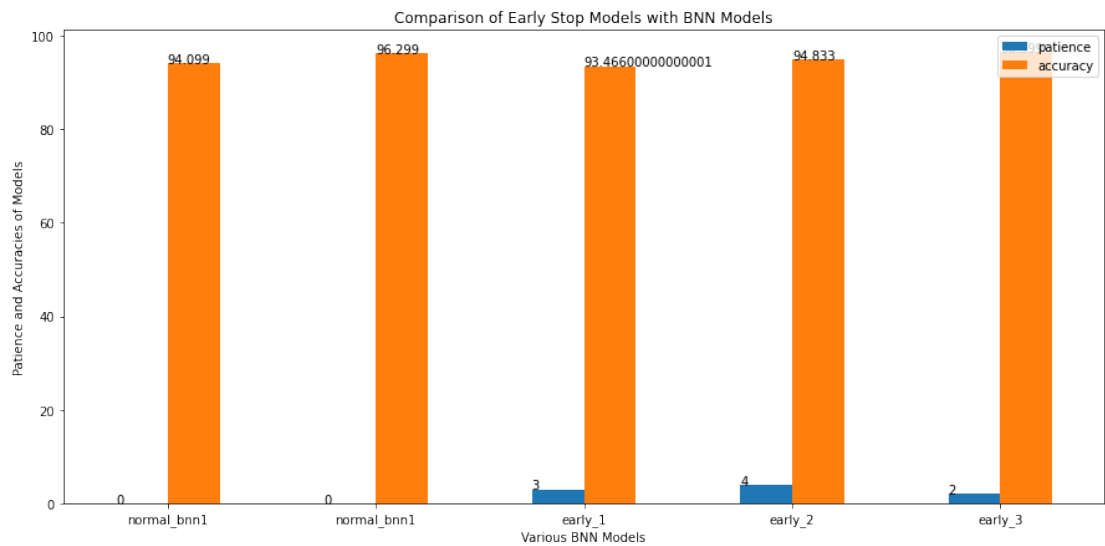


Figure 4.28: Comparison of all Early stop models with BNN Models
Comparison of the number of early stop patience value and accuracy's for these models.

In Early stop BNN models, the model with patience=2 is performing better when compared to other two early stop models.

CHAPTER 5

SUMMARY AND CONCLUSION

In this chapter, we discuss the summary of this thesis or report and the conclusion of this project.

Expert system has many advantages compared to its limitations. So to overcome its limitations, we can use various practical implementation techniques of Bayesian neural networks as a part of its inference engine which approximates them yet gives better results. Some of the approaches implemented in this work are, MC Dropout, Early Stopping.

A different set of Bayesian Neural Network models are built using various techniques and variations in different parameters. All these models are used in the web application tool which is built for developing an expert system. The tool implemented in this project has a prediction page where we can get the prediction from any model with given input variables. It also has an expert system page where a user can answer the given questions and get the answers from the tool which uses the models to answer the questions.

The dataset taken is for the task of Predictive Maintenance which helps in the identification of the most probable root cause of a problem or failure and to give a solution to rectify it. This way, the frequency of maintenance can be reduced by having maintenance only when certain conditions are encountered and when it is necessary instead of scheduled maintenance. This results in higher efficiency of the machine with low maintenance cost over a long period of time.

Besides these, there are many other advantages of using predictive maintenance for machines in industries and other businesses across different domains. Some of these advantages are reduced maintenance costs on machines, more availability of machines with reduced downtime of a machine, higher efficiency and safety and with a longer life span.

Future Scope:

This work can be extended for various domain applications. An Expert system can be developed for the domains like Health care, software bug detection, predicting the machine's remaining lifetime based on usage, etc. It can also be extended by adding more number of questionnaire to answer by the user for better understanding which will lead to better performance. This can further be extended or improved by having more machine learning algorithms for a better comparison of results and also by having more data.

We can also extend this work to make an Expert Shell System where we can use this as a building block or as an environment for developing any Expert System, just by changing the knowledge base for the specific system. They can be used for building various applications just from the template like an expert shell system which is a generalized one which helps in building any domain-specific expert system for better decision-making.

REFERENCES

1. **M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng** (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. URL <https://www.tensorflow.org/>. Software available from tensorflow.org.
2. **H. Abd-Elhamid, A. Javadi, A. Negm, A. Elalfi, and T. Owais** (2011). Development of an expert system for maintenance and repair of masonry barrages. *Intelligent Computing in Engineering–ICE08*, 212–221.
3. **A. A. Adekunle, P. P. Ikubanni, and O. O. Agboola** (2018). An expert system for automobile repairs and maintenance. *Mindanao Journal of Science and Technology*, 16(1).
4. **O. B. Akinluli, V. A. Balogun, and T. M. Azeez** (2015). Development of an expert system for the repair and maintenance of bulldozer’s work equipment failure. *International Journal of Scientific & Engineering Research*, 6(6), 1–14.
5. **K. Bykov, M. M.-C. Höhne, A. Creosteanu, K.-R. Müller, F. Klauschen, S. Nakajima, and M. Kloft** (2021). Explaining bayesian neural networks. *arXiv preprint arXiv:2108.10346*.
6. **D. T. Chang** (2021). Bayesian neural networks: Essentials. *arXiv preprint arXiv:2106.13594*.
7. **Z. M. Çınar, A. Abdussalam Nuhu, Q. Zeeshan, O. Korhan, M. Asmael, and B. Safaei** (2020). Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0. *Sustainability*, 12(19), 8211.
8. **J. V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. D. Hoffman, and R. A. Saurous** (2017). Tensorflow distributions (tensorflow probability). *CoRR*, abs/1711.10604. URL <http://arxiv.org/abs/1711.10604>.
9. **D. Dua and C. Graff** (2017). UCI machine learning repository. URL <http://archive.ics.uci.edu/ml>.
10. **N. Hassen and I. Rish** (2021). Approximate bayesian optimisation for neural networks. *arXiv preprint arXiv:2108.12461*.
11. **L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun** (2020). Hands-on bayesian neural networks – a tutorial for deep learning users.

12. **V. Kontargyri, C. Tsirekis, S. Kabanaros, A. Sakarellos, A. Moronis, N. Kolliopoulos, and S. Kaminaris**, An expert system for fault diagnosis, repairing and maintenance of electrical machines. *In Proceeding of the 6 th WSEAS International Conference on Applications of Electrical Engineering*. Citeseer, 2007.
13. **N. Mourdoukoutas, M. Federici, G. Pantalos, M. van der Wilk, and V. Fortuin** (2021). A bayesian approach to invariant deep neural networks. *arXiv preprint arXiv:2107.09301*.
14. **S. Nguyen, D. Nguyen, K. Nguyen, N. Ho, K. Than, and H. Bui** (2021). Structured dropout variational inference for bayesian neural networks.
15. **V. Rakesh and S. Jain**, Efficacy of bayesian neural networks in active learning. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.
16. **E. A. Simeón, A. J. Álvarez, and R. R. Gudwin**, An expert system for fault diagnostics in condition based maintenance. *In ABCM Symp. Ser. Mechatron*, volume 4. 2010.
17. **Streamlit** (). Streamlit framework. <https://streamlit.io/>.