Text

Description automatically generated

**UNIVERSITI TEKNOLOGI MARA**

**ASSIGNMENT 3**

**DESIGNING A FORCASTING MODEL FOR TIME SERIES DATA**

**RAMADAN YACIN IBRAHIM**

(SID- 2021935561) (GRP- CS7073A)

**FOR:**

**PROFESOR MADYA ZAIDAH BINTI IBRAHIM**

**2022**

1. **Introduction**

Recurrent neural networks (RNNs) are a form of neural network in which the results of one step are fed into the next phase's computations. Traditional neural networks have inputs and outputs that are independent of one another, but there is a requirement to remember the previous words in situations when it is necessary to anticipate the next word in a phrase. As a result, RNN was developed, which utilised a Hidden Layer to resolve this problem. The Hidden state, which retains some information about a sequence, is the primary and most significant characteristic of RNNs.

RNNs have a "memory" that retains all data related to calculations. It executes the same action on all of the inputs or hidden layers to generate the output, using the same settings for each input. In contrast to other neural networks, this minimises the complexity of the parameter set.

1. **Materials and Methods**

**Dataset**

Time series data, also referred to as time-stamped data, is a sequence of data points indexed in time order. Time-stamped is data collected at different points in time. These data points typically consist of successive measurements made from the same source over a time interval and are used to track change over time.

Graphical user interface, application, table, Excel

Description automatically generated

Fig 1 : data csv file (SungaiSediliKecil)

1. ***IDE and Programming language***

In this assignment we are using Python IDLE and programming language. Python is a programming language that supports the creation of a wide range of applications. Developers regard it as a great choice for Artificial Intelligence (AI), Machine Learning, and Deep Learning projects.

1. ***Machine learning Algorithm***

***Recurrent neural networks (RNNs)***

As seen by the unrolled RNN in the image above, RNNs function by utilising backpropagation across time (BPPTT). Weights for the most recent and preceding input are updated in this way. By propagating the mistake from the most recent time step to the first, the weights are modified. The error for each time step is calculated as a consequence.

Long-term dependencies are not something that RNNs can handle. Vanishing gradients and bursting gradients are two significant issues that appear in extremely long time steps.

***Text

Description automatically generated***

Fig 1: code

Diagram

Description automatically generated

Fig 2: workflow

1. **Experimentation (Train and testing )**

Text

Description automatically generated

Fig 3 : code of RNN

Text

Description automatically generated

Fig 3: RNN performance **Accuracy : 0.96**

The accuracy obtained with RNN is 0.96. secondly, we have performed the test on with the same algorithm by different hyperplane parameters. The epoch was set to 1 this time. And the loss obtained is 0.967

Graphical user interface, text

Description automatically generated

Fig 3: RNN performance **Accuracy : 0.97 (epoch =2)**

The accuracy obtained with RNN has become 0.97 after increasing the epoch.

1. **Result and Discussion**

|  |  |  |  |
| --- | --- | --- | --- |
| **Epoch = 1** | | **Epoch =2** | |
| **0.96** | **0.096** | **0.97** | **0.094** |

Table 1 : Accuracy of ML algorithms

After increasing the epoch from 1 to 2, we discovered that increasing the epoch has increased the accuracy and decreased the loss. in thIS case of recurrent neural networks, the output from the previous steps is fed into the input of the current state. For instance, to predict the next letter of any word or to predict the next word of the sentence, there is a need to remember the previous letters or the words and store them in some form of memory.

1. **Conclusion**

The RNN algorithms have been programmed in Pythonusing scikit-learn library in IDLE environment. After an accurate comparison between our models, we foundthat it achieved a higher efficiency of 0.97, compared to with epoch 1. The accuracy obtained with RNN is 0.96. secondly, we have performed the test on with the same algorithm by different hyperplane parameters. The epoch was set to 1 this time. And the loss obtained is 0.967.