Comparing the predictive capabilities of **Artificial Neural Networks** and **Support Vector Machines** in forecasting geographical features of traffic accidents and the **number of injuries**.

Research Question: To what extent is a deep learning **Artificial Neural Network (ANN)** more suitable than **Support Vector Machines (SVM)** at predicting the **number of injuries** in traffic accidents in Uiwang City, South Korea?

Subject: Computer Science
Wordcount: 3968

Contents

Introduction......2
Background Information......4
1. Support vector machine......4
2. Artificial Neural Network......5
3. Python library......8
3.1 Sklearn library......9
3.2 Pandas library......9
Methodology......10
Variable......13
Data Presentation......22
Data analysis......22
1. Result......22
2. Different test data size......23
Limitations of Investigation......25
Conclusion......27
Works Cited......28

Introduction

Globally, with the increasing highway mileage and vehicle quantity, the potential of traffic accidents is also increasing, resulting in a series of unfavorable lives for people. Injuries from a traffic accident are deadly as they can be small bruises to death on the spot. According to Korea's National Indicators System, 560,000 people in South Korea lost their lives in road traffic accidents in 2021(OECD KOREA). Hence, once the accident happens, predicting injuries or any casualties and prioritizing the list is thought to play a key role in saving people. These critical injuries are time-dependent, and as soon as the ambulance arrives at the accident scene with a potential injury, the more chance casualties can recover well. According to MedicineNet.com, this gets really important, especially in cases of traumatic injuries, as this golden hour plays an essential role in deciding the mortality of the patient(Pallavi Suyog Uttekar, MD).

Predicting the number of injuries and prioritizing the most dangerous situation can develop and improve the Emergency Medical Service (EMS) system by reducing the wasted time in deciding where to go. Hence, this investigation will compare two machine learning algorithms for the accuracy and precision of their prediction to know what algorithm is more suitable in an emergency situation. Both Support Vector Machines (SVM) and Artificial Neural Networks (ANN) are chosen because they both are well known for predicting future events based on the data. The two machine learning approaches were chosen because of their accuracy in regression analysis with the following reasons behind.

SVM is renowned for future event prediction by transforming input data into a high-dimensional feature space. It excels in binary classification tasks, such as forecasting injury counts with associated data(baeldung). ANN, a prevalent data mining technique, excels as a nonlinear estimator, capturing intricate input-output relationships, applicable in domains like traffic prediction(Kalita). This question will be answered by testing both classifiers on Google Collab with the same data point and comparing their accuracy and precision.

Background Information

1. Support vector machine

SVMs are employed for classifying two distinct data groups by creating hyperplanes that segregate the groups based on patterns. This process forms a model that can classify new examples. SVMs necessitate labeled data for training, which organizes training data into distinct groups in space(Gandhi). Figure 1 provides a visual illustration of 2D and 3D kernels. The decision surface is formed along the surface in three dimensions.

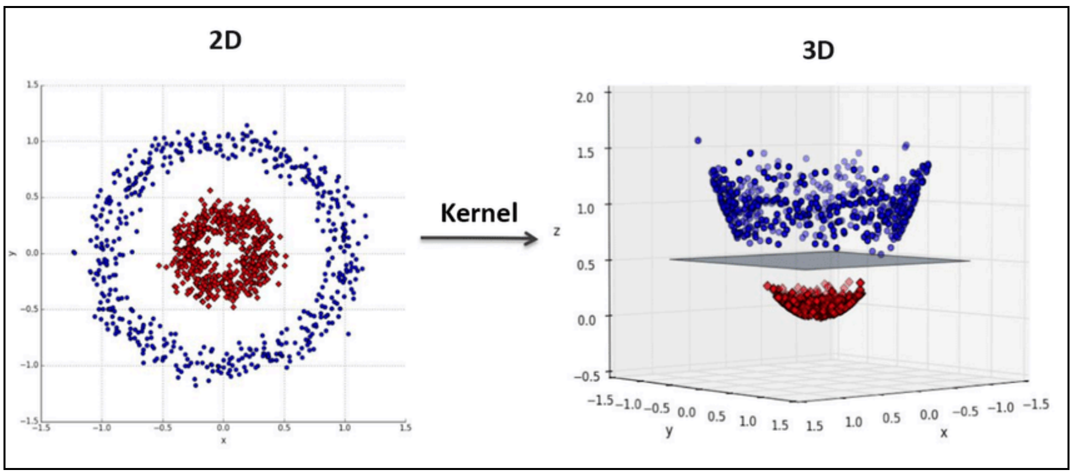

Fig. 1. "Non-Linear Classifier Using Kernel Trick [16]." *ResearchGate*, ResearchGate, 2020.

To increase the accuracy in SVM, optimizing margin takes an important role. Optimizing margin refers to the maximum width of the side that runs parallel to the hyperplane.SVM was potentially designed for binary classification problems; however, with the rise in computationally intensive multiclass problems, several binary classifiers are constructed and combined to formulate SVMs that can implement synch multiclass classification through binary means(Avinash Navlani).

Figure 2 visualizes what a linear hyperplane would look like. The red line is the decision line, and the parallel line represents the maximum spacing between the data to optimize the margin.

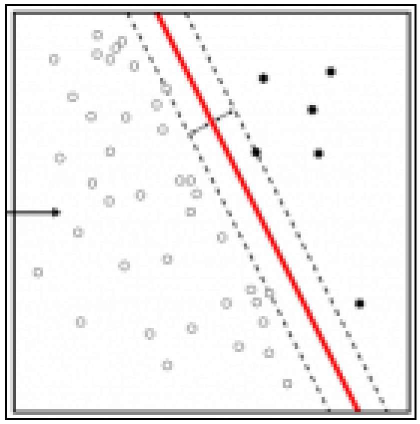

Fig. 2. bwv549. “The Shunning Key: Whom Latter-Day Saints Shun and Why.” *A Careful Examination*, 2023

Kernel methods play an important role in SVM models as they transform the data features by employing kernel functions to map complex datasets to higher dimensions. This takes place in a manner that makes data point separation easier; the function simplifies the data boundaries for non-linear problems by adding higher dimensions to map complex data points(“All You Need to Know about Support Vector Machines”).

2. Artificial Neural Network

To comprehend the concept of an ANN, it's crucial to familiarize oneself with the notions of inputs, weights, and bias. Inputs represent the collection of values we aim to use in predicting an output value and can be thought of as the characteristics or attributes within a dataset. Weights are actual numerical values linked to each input or feature, serving to indicate the significance of that specific feature in forecasting the ultimate output. Lastly, bias plays a role in adjusting the activation function, shifting it either to the left or right, somewhat akin to the y-intercept in a linear equation.

ANN is a computational model that mimics the way nerve cells work in the human brain. It uses learning algorithms that can independently make adjustments or learn as they receive new input. This neural networking system has various layers that are interconnected. The first layer consists of input neurons and sends data to the deeper layers which in turn will send the final output data to the last output layer. Each layer acts as both an input and output layer which allows the ANN to understand more complex objects. When the connections between layers are highly weighted, the greater influence one unit has on another.

In order for an ANN to acquire knowledge, it needs a substantial amount of data known as a "training set." During this training process, the machine's generated output is compared to a human-provided description of what the machine should ideally recognize or understand. If the machine's output matches the human description, it is considered validated(sampath kumar gajawada). However, if it produces an incorrect result, it employs a technique called "backpropagation" to make adjustments to its learning. Backpropagation involves going back through its layers of processing to fine-tune the mathematical equations it uses for learning. This iterative process helps the machine improve its performance and learn more accurately over time.

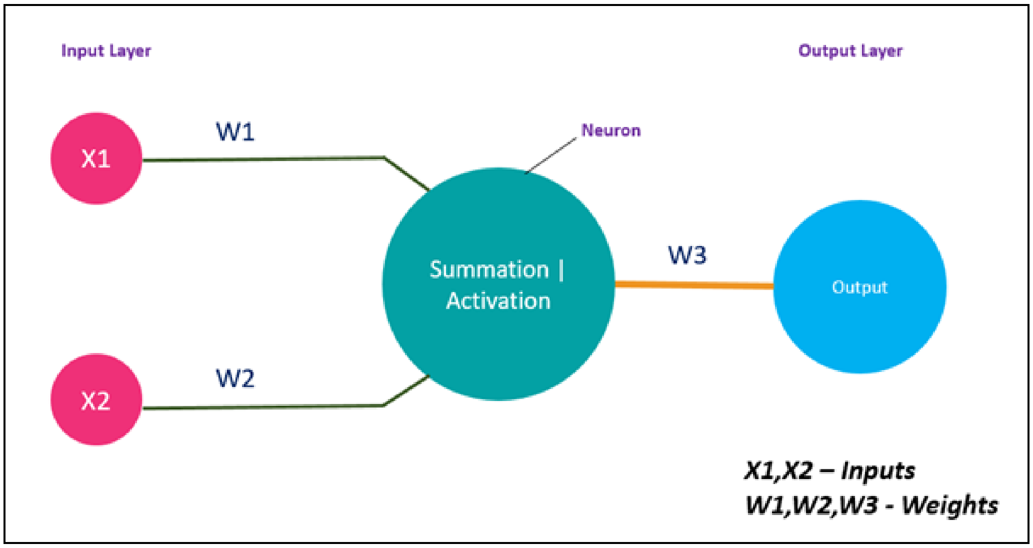

Fig. 3. sampath kumar gajawada. "The Math behind Artificial Neural Networks - towards Data Science." *Medium*, Towards Data Science, 2019

Figure 3 is the visualization of ANN. Going from the left-hand side, it shows the input layer, hidden layer and output layer. It is not expressed in the figure, but multiple layers are possible for the hidden layer but not the input and output layers. Figure 3 also represents a perceptron in ANN which is a simple artificial neuron having an input layer and an output layer. From this simplification, it is easier to understand the math behind this algorithm. The summation function combines the weights and inputs together and calculates their sum. Fig.4 has a further explanation of different summation functions.

Weighted total	$NET = \sum x_i . w_i$	Inputs and weight values are multiplied. The calculated values are added to each other.
Multiplication	$NET = \prod x_i . w_i$	Inputs and weight values are multiplied. The calculated values are multiplied.
Maximum	$NET = Max(x_i . w_i)$	Inputs and weight values are multiplied. The highest calculated value is taken.
Minimum	$NET = Min(x_i . w_i)$	Inputs and weight values are multiplied. The lowest calculated value is taken.
Incremental total	$NET_k = NET_{k-1} + \sum x_i . w_i$	Weighted total is calculated. The previous weighted total is calculated.

Fig. 4. “The Summation Functions Used in Artificial Neural Networks...” *ResearchGate*, ResearchGate, 2018

Activation function is a function to convert the weighted sum of input signals of a neuron into the output signal(it serves as input to the next layer). Rectifier Linear Unit(ReLU) is one of the activation functions that will be used in this investigation. This particular function operates like a switch, where it lets the input through if the input is positive, otherwise it produces a zero output. ReLU has gained popularity as the default activation function in various neural network applications due to its ease of training and its tendency to deliver superior performance in models.

3. Python library

A Python library is like a toolbox full of tools that are connected in some way. It holds sets of code that you can use over and over again in various programs. This makes programming in Python easier and more convenient for the person writing the code. These libraries are especially important in areas like Machine Learning, Data Science, and Data Visualization, where they play a big part in helping with various tasks(“Libraries in Python”).

3.1 Sklearn library

Sklearn, also known as scikit-learn, is a free software machine learning library for the Python programming language. In this investigation, the sklearn library will be used most of the time due to its usefulness in machine learning. First, it "provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction through a consistent interface in Python"("Amarta Karya"). With SVM and ANN the library contains code for SVM and ANN, which includes all the functions and calculations needed("Scikit Learn - Introduction").

Codes/functions stored in Sklearn library:

- from sklearn import svm
- from sklearn import metrics
- clf = svm.SVC(kernel='linear')
- clf.fit(X_train, y_train)
- y_pred = clf.predict(X_test)
- print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

("Scikit Learn - Introduction")

3.2 Pandas library

The Pandas library is specifically designed for dataset manipulation. It offers a suite of tools for data analysis, cleaning, exploration, and manipulation. It empowers users to perform statistical analysis on large datasets, transforming them into a more organized and comprehensible form. In this study, Pandas play a crucial role in structuring data for efficient use by other algorithms(“Pandas Introduction”).

Methodology

In this investigation, the initial step involves acquiring a suitable dataset for training our model by comparing two algorithms. A suitable dataset should encompass various essential elements for estimating accident casualties. These critical dataset attributes include time-related information, the count of fatalities, the number of casualties, accident types (such as individual versus vehicular involvement), details about the assailant's vehicle, and potential violations of legislative regulations by the perpetrator. Given that this dataset initially appears as strings (sequences of characters), converting them into integer representations is a crucial step that enhances the computational efficiency of the code.

The data for this inquiry originates from data.go.kr, an authoritative repository providing comprehensive information on contemporary traffic accidents in Uiwang City. The dataset contains a wide range of details, including urban and territorial designations, event timestamps, occurrences on the calendar, differentiation between daytime and nighttime events, types of accidents, overarching classifications, divisions of accident types, and instances of legal violations. This dataset covers the period from January 1, 2012, to December 31, 2021, with annual updates being the standard practice(도로교통공단).

Type of accident		Road type	
Car vs. Car	1	intersection	1
Car itself	2	single track	2
Car vs. Person	3	other	3
Specific type of accident		Perpetrator classification	
collision	1	car	1
Collision of structures	2	a motorized bicycle	2
stratified collision	3	truck	3
Crossing	4	van	4
head-on collision	5	two-wheeled vehicle	5
roll over	6	construction machinery	6
others	7	special car	7
on the sidewalk	8	personal means of transportation	8
edge of a road	9	a four-wheeled motorcycle	9
driveway in traffic	10	other	10
capsize	11	bike	11
departure from the road	12	agricultural machinery	12
departure from the road and fall	13	unknown	0
collision with startionary vehicle	14	victim classification	
collision on it's sternway	15	none	0
offense against the law of the perpetrator		Pedestrian	1
Failure to comply with the duty of safe driving	1	car	2
other	2	a motorized bicycle	3
signal violation	3	truck	4
Violation of the way of intersection	4	van	5
invasion of the centerline	5	two-wheeled vehicle	6
violation of pedestrian protection obligations	6	construction machinery	7
fall	7	special car	8
Failure to secure a safe	8	personal means of transportation	9
offense against the law of the perpetrator		a four-wheeled motorcycle	10
Failure to comply with the duty of safe driving	1	bike	11
other	2	agricultural machinery	12
signal violation	3	other	13
Violation of the way of intersection	4	unknown	14
invasion of the centerline	5	weekday	
violation of pedestrian protection obligations	6	Mon	1
fall	7	Tues	2
Failure to secure a safe	8	Wed	3
day / night		Turs	4
day	1	Fri	5
night	2	Sat	6
		Sun	7

Table 1: Screenshot of organization of dataset (designed by candidate).

Subsequently, this dataset is transformed into a CSV file, simplifying its importation for model training purposes. To enhance the algorithm's efficiency, Table 1 arranges the dataset as integers, reducing storage requirements and reducing the amount of work the algorithm has to do. This data transformation could lead to quicker results, as integer data types consume less storage than strings.

The Support Vector Machine (SVM) learning code is then deployed through IDE (integrated development environment). SVM training follows a systematic procedure aiming to create a model capable of precise classification or prediction. Python libraries are employed to split the data into training and testing sets and especially, sklearn library is a perfect library for SVM to code. The next step involves training the classifier with the training data and making predictions on the test data. The algorithm concludes by presenting insightful information about accuracy while utilizing the trained classifier to make predictions(Chat GPT, 2023).

Training an ANN on a dataset involves creating an effective structure and optimizing its key parameters. Here's how it works: when the neural network receives input, it generates an output. During the learning phase, each input is accompanied by a label that indicates the expected output. If the network's guess aligns with the label, it maintains its current configuration. However, if the output doesn't match the label, the network adjusts its weights, which are the only elements that change during this learning process. Think of this as fine-tuning buttons to improve the network's accuracy. Ultimately, the primary aim is to ensure that our predictions are accurate and that our categories align with the data we have("How to Build and Train an Artificial Neural Network").

Variable

Independent variable	**Type of classifier:** - Support Vector Machine classifier - Artificial Neural Network classifier
Dependent variable	**Accuracy** - In the sklearn library, there is an encoded command term that calculates the accuracy of the algorithm by "comparing the sets of labels predicted for a sample must exactly match the corresponding set of labels in original data." ("Sklearn.metrics.accuracy_score"). The result will come out as a decimal between 0 and 1, which can be read as a percentage. **Precision** - Average error with the prediction and original data.
Controlled variable	**Testing data size:** Both data that are trained have to be large enough for the model to find a pattern. For example, ANN performs better with large data, while SVM can also work with small amounts of data. **Data set:** Each data set that the model is based on has to be identical so that the result is controlled. **Ratio of training data and testing data:** When the model is trained, it learns in two ways. First, it learns the pattern of the data from the training data, and it tests its function on testing data. Through the trial of testing data, it increases the accuracy of learning what is wrong and what is right. In this investigation, the testing data will be 25% of the data and

	training data will be 75%. According to scholarworks.utep.edu, “empirical studies show that the best results are obtained if we use 20-30% of the data for testing and the remaining 70-80% of the data for training”.

Procedure in steps:

SVM:

1. Find the appropriate dataset for the testing. The format must be in CSV form, and it must contain a variable to predict. CSV form stands for comma-separated values, as the values are separated by commas in a plain-text file. These CSV files allow the developers to import into a spreadsheet or another storage database, regardless of any software like colab that is used in this investigation. In addition, it is also effectively used when organizing large amounts of data, such as the data on car accidents from 2012 to 2021(“What Is a .CSV File and What Does It Mean for My Ecommerce Business?”).

2. To set up the data, first the file has to be uploaded to the IDE as a CSV file. Next, the feature names have to be defined for the dataset column to indicate input and output data. Hence, the name of the features must match the name of the column of the original data. This is the case in figure 5, the code is translated into English so that it is easier to understand. (The original data is written in Korean so the code also had to be in Korean.) (see figure 5)

```
form google.colab import files
Uploaded = files.upload()

Import pandas as pd
df2 = pd.read_csv(io.BytesIO(uploaded['Proceesed_data_ -
DeathTrafficAccidentInformation(2021) (1).csv']))

feature_names = ['Date of occurrence', 'day and night', 'day and day',
'number of deaths', 'number of injured', 'type of accident_large
classification', 'type of accident', 'violation of the perpetrator's law',
'type of road', 'type of perpetrator', 'type of victim']

df2.columns = feature_names
```

Fig. 5. Screenshots of code uploading the file and signing features

3. To process the raw data for the SVM classifier, the data has to be split into testing data and training data to efficiently train the model as features (X) and target labels(Y). For this example, the number of 'number of injured' is the target data to predict so that the column is assigned to X and Y, for this step sklearn.model_selection library is used to split the data for training and testing into 1:3 ratio. (see figure 6)

```
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import metrics
X = df2.drop('number of injured', axis=1)
y = df2['number of injured']

X_tr, X_tst, y_tr, y_tst = train_test_split(X, y, test_size=0.25,
random_state=109)

clf = svm.SVC(kernel='rbf')

clf.fit(X_tr, y_tr)
```

Fig. 6. Screenshot of code processing the data

4. Lastly, after all the calculations of accuracy and precision by sklearn.metrics library function and a simple loop, the resulting values have to be printed("Sklearn.metrics.accuracy_score")(see figure 7).

```
y_pr = clf.predict(X_tst)

accuracy = metrics.accuracy_score(y_tst, y_pr)
print("Accuracy:", accuracy)

Count = 0
total_error = 0
for V in range(len(X)):
prediction = clf.predict([X.iloc[V]])
real_value = y.iloc[V]
if prediction != real_value:
total_error += abs(prediction - real_value)
Count += 1

if Count != 0:
print("Average error: ",total_error / Count)
else:
print("No errors found.")
```

Fig. 7. Screenshot of code outputting accuracy, average error and total error count

ANN:

1. After uploading the datasets to the colab like the previous model, the data has to be processed. Like the previous algorithm, the data has to be split into testing data and training data with the right ratio of 1:3.

2. When data is loaded, prepare the data to extract the target variable ‘number of injured’ into the Y array and other values in the X array(see figure 8).

```
# Extract the target variable (y)
y = dataset['number of injured'].values

# Extract the features (X)
X = dataset.drop(['number of injured'], axis=1).values

# Data preprocessing - Standardization
sc = StandardScaler()
X = sc.fit_transform(X)
```

Fig. 8. Screenshot of code assigning features

3. The data needs to be split into training and testing sets to effectively evaluate the algorithm. Utilizing the 'train_test_split' function from scikit-learn, partition the data into training and testing sets, allocating 25% of the data to the testing set. This variable is set to be the same with SVM(see figure 9).

```
# Splitting the dataset into the Training set and Test set

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=0)
```

Fig. 9. Screenshot of code processing data

4. Subsequently, proceed to initialize the ANN Model. Begin by establishing a sequential model, which involves the linear arrangement of layers using Keras (A free-to-use deep learning platform created to simplify the process of constructing and testing artificial

neural networks, catering to both researchers and developers). Start by adding an input layer, followed by the first hidden layer consisting of six units. This layer will utilize the Rectified Linear Unit (ReLU) activation function. The input dimension for this layer is determined by the number of features present in the dataset. It is important to note that ReLU, short for Rectified Linear Unit, is selected for its computational efficiency, primarily involving straightforward thresholding operations. Following this, introduce a second hidden layer configured identically to the first one. Lastly, incorporate the output layer, featuring a single unit, which is commonly employed for regression purposes and does not employ any activation function(see figure 10).

```
# Initialize the Artificial Neural Network for regression
regressor = Sequential()

# Add the input layer and the first hidden layer
regressor.add(Dense(units=6, kernel_initializer='uniform',
activation='relu', input_dim=X.shape[1]))

# Add a second hidden layer
regressor.add(Dense(units=6, kernel_initializer='uniform',
activation='relu'))

# Add the output layer
regressor.add(Dense(units=1, kernel_initializer='uniform')) # No
activation function for regression
```

Fig. 10. Screenshot of code training ANN

5. Compiling and fitting the model is an important process to configure the learning process before training. It tells the computer to set up its model for the prediction. The aim of the

algorithm is to be smart and learn from its mistakes so that ‘compile’ sets up the model. Under the code, a tool called “adam” is used to learn and to pay attention to how wrong the predictions are. Fitting the data refers to teaching the computer how to make better predictions with examples of correct answers(Keras)(see figure 11).

```
# Compile the ANN for regression
regressor.compile(optimizer='adam', loss='mean_squared_error') # Use mean squared error for regression

# Fit the ANN to the Training set
regressor.fit(X_train, y_train, batch_size=10, epochs=100, verbose=0)
```

Fig. 11. Screenshot of code compiling and fitting the data

6. Use a trained neural network model (regressor) to predict target values ‘y_pred’ from the testing data. Then, convert the predicted floating-point values to integers, making it ready for further analysis or evaluation. Using ‘accuracy_score’ code to calculate the difference between the original value and the prediction from the trained model and print it out. Similar to SVM, it displays the average uncertainty per 100 data and the count of the wrong prediction per 100 data.

7. Next, the accuracy and the prediction have to be printed out. Using sklearn library, compare the predicted value and original value accuracy(see figure 12).

```
# Predicting the test results
y_pred = regressor.predict(X_test)

# Convert predictions to integers (assuming you want integer predictions)
y_pred = y_pred.astype(int)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

# Display predictions for each value
Count = 0
total_error = 0
for V in range(len(X)):
      prediction = clf.predict([X.iloc[V]])
      real_value = y.iloc[V]
      if prediction != real_value:
            total_error += abs(prediction - real_value)
            Count += 1

if Count != 0:
      print("Average error: ",total_error / Count)
Else:
      print("No errors found.")
```

Fig. 12. Screenshot of outputting accuracy, average error and error count

Data Presentation

	SVM	ANN
Accuracy (%)	79.4	78.5
Average uncertainty of prediction	2.031	1.863
Total error counted	334	339

Table 2: Data of accuracy, average uncertainty and error counted

Data analysis

1. Result

The dataset provides insights into the accuracy and uncertainty of SVM and ANN classifiers. Uncertainty in error prediction, reflecting precision, is a crucial metric for evaluating these models. SVM demonstrates 79.4% of accuracy in predicting the right data, in comparison to 78.5% accuracy of ANN, signifying a higher overall correct prediction rate(see Table 2).

This observation is substantiated by error counts, with SVM boasting an average error count of 23.9, while ANN registers 24.2. However, it is imperative to note that ANN showcases a lower average uncertainty, with a precision level denoted by an average uncertainty of 1.863, outperforming SVM's 2.031(see Table 2). This distinction highlights that ANN's predictions are in closer proximity to actual outcomes, indicating a superior level of precision.

In summation, SVM exhibits a strength in accuracy, rendering it well-suited for certain applications. Conversely, ANN's reduced uncertainty renders it advantageous for scenarios necessitating a high degree of precision, such as precise financial forecasting or medical diagnostics. The choice between prioritizing accuracy or precision hinges upon the specific demands and objectives of the predictive tasks.

2. Different test data size

Type of classifier\size of test data	0.1	0.5	1.0
SVM	80.9%	77.6%	100%
ANN	78.4%	75.1%	0%

Table 3: Different size of test data and accuracy for each classifier

However, as the accuracy of both classifiers is adjacent to each other, differing by only 2%, the accuracy can be influenced by changes in the size of the test data, impacting the results of the investigation. Hence, Table 3 shows a different result when the size of the test data changes the accuracy of each classification. Overall, SVM shows higher accuracy over ANN by roughly 2% and when the whole data is used for testing, SVM shows 100 percent accuracy and ANN shows 0% accuracy(see Table 3). Nevertheless, when the whole data is used in testing, there are some consequences.

The perfect score for SVM is only caused by the logic error in the code. The classifier is not only tested on data it has seen before, it is tested on the exact same data that was used for training. This results in perfect accuracy because the model has already seen the test data during training and can predict it correctly. It is essential to split the data into training and testing sets to assess the model's ability to generalize to new, unseen data. For ANN, when 100% of data is used for testing, it means there are no data points left in the training set. This results in the model not being trained at all, which reflects the accuracy of 0 as it has no knowledge to make predictions. Therefore, accuracy for 1.0 test data size does not reflect the model’s true performance on new data.

In conclusion, the overall trends from both Tables 2, and 3 consistently demonstrate that the SVM exhibits superior accuracy compared to the ANN, which aligns with the results in Table 2. This data supports the conclusion that, under these conditions, SVM is the more accurate classifier. However, it's important to note that the choice between SVM and ANN should consider the specific requirements and objectives of the predictive tasks, as well as the size of the test data.

Limitations of Investigation

Limitations of the investigations include the following:

1. The data that was used to train the data was collected from 2012 to 2021. This old data might not be relevant to predict the current accidents using the past database. This could be largely impacted by the development of technology. For example, the self-driving car might have reduced the injuries which is a factor that the algorithm can not detect.

2. Not all features were used to predict the data. Some of the features might not be related to the prediction. If there is an unrelated feature analysis to predict a value, that feature might have given the wrong interpretation to the algorithm and decrease the accuracy and precision of the prediction.

3. In reality, it is hard to collect this data in an emergency situation. The reporter might not know the exact type of road, type of vehicle, and the cause of the accident. When the algorithm lacks a few features, the prediction of the classifier becomes useless as it is not based on the right dataset (Kopf). In addition, asking those questions to the reporter could be unnecessary and meaningless if the injury is observable without additional features.

4. For this experiment, as the sample size dataset was tested, there is a limitation on the size of the dataset. Both of these datasets could have improved on their prediction, especially ANN which works better with large data and might have shown an improved accuracy rate (Haruna Chiroma et al.).

5. The dataset that was the base of the prediction does not show the physical features of the victim. For example, the posture of the victim determines how much impulse they received in the specific area the person got damaged because the chance of survival differs for each part of the body. In addition, muscularity, osteoporosis, and other health-related illnesses run as important factors in injuries and survival.

Conclusion

This experiment aimed to compare SVM and ANN to predict future events, especially the number of injured persons in car accidents by implementing outer features. With the dataset that provides features of ‘Date of occurrence', 'Day and night', …, SVM demonstrates an accuracy of 79.4%, while ANN achieves 78%, with similar prediction capabilities. However, in terms of precision, ANN's uncertainty rate of 1.863 is notably lower than SVM's 2.031. Within the topic of predicting the future event, accuracy is more valuable than precision as the value has to match with the future, yet in this situation, having an aim to go save people, it needs more precision than accuracy to handle injured people. This means that ANN is more suitable classier for this investigation. In addition, the number of ambulances sent out could be adjusted at any time so that the accuracy of the prediction is not pressured. This prediction will be helpful as the ambulance could save their time coming to the place of the accident. An ambulance has a capacity for one person, when the number of injured people is predicted, it would easily sort out how many ambulances to be dispatched. However, this experiment does not fully represent a real life situation and there are some limitations that make it hard to predict in real life. Some major limitations are the lack of a dataset as it did not have enough data to train the classifier and the unpredictability of emergency situations. People might panic, or the reporter does not have the ability to provide other information to predict.

Works Cited

"6 Repeat Step 1 to 5 and Update the Weights after Each Observation Reinforcement - USC410H | Course Hero." *Coursehero.com*, 2022, www.coursehero.com/file/p3i4f68/6-Repeat-step-1-to-5-and-update-the-weights-after-each-observation-reinforcement/?cv=1. Accessed 27 Sept. 2023.

"All You Need to Know about Support Vector Machines." *Spiceworks*, 29 Sept. 2022, www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/. Accessed 21 Sept. 2023.

Fan, Shuzhan. "Understanding the Mathematics behind Support Vector Machines." *Shuzhan Fan*, 7 May 2018, shuzhanfan.github.io/2018/05/understanding-mathematics-behind-support-vector-machines/. Accessed 23 Sept. 2023.

---. "Understanding the Mathematics behind Support Vector Machines." *Shuzhan Fan*, 7 May 2018, shuzhanfan.github.io/2018/05/understanding-mathematics-behind-support-vector-machines/. Accessed 3 Oct. 2023.

Haruna Chiroma, et al. "Progress on Artificial Neural Networks for Big Data Analytics: A Survey." *IEEE Access*, vol. 7, 1 Jan. 2019, pp. 70535–70551, ieeexplore.ieee.org/document/8531611, https://doi.org/10.1109/access.2018.2880694. Accessed 19 Sept. 2023.

"How to Build and Train an Artificial Neural Network." *Nickmccullum.com*, 2023, www.nickmccullum.com/python-deep-learning/artificial-neural-network-tutorial/. Accessed 15 Oct. 2023.

"How to code SVM and ANN" prompt. ChatGPT, GPT-3, OpenAI, 15 Aug. 2023, chat.openai.com/chat.

Keras, in. "What Do "Compile", "Fit", and "Predict" Do in Keras Sequential Models?" *Data Science Stack Exchange*, 24 Feb. 2019, datascience.stackexchange.com/questions/46124/what-do-compile-fit-and-predict-do-in-keras-sequential-models. Accessed 28 Sept. 2023.

Kopf, Dan. "This Is How Computers "Predict the Future."" *Quartz*, Quartz, 5 Sept. 2018, qz.com/1261817/predictive-algorithms-are-not-all-that-complicated. Accessed 19 Sept. 2023.

Kuruva Satya Ganesh. "Weights and Bias in a Neural Network | towards Data Science." *Medium*, Towards Data Science, 24 July 2020, towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f. Accessed 21 Sept. 2023.

"Libraries in Python." *GeeksforGeeks*, GeeksforGeeks, 16 Oct. 2021, www.geeksforgeeks.org/libraries-in-python/. Accessed 30 Sept. 2023.

Marr, Bernard. "What Are Artificial Neural Networks - a Simple Explanation for Absolutely Anyone." *Forbes*, 12 Oct. 2022, www.forbes.com/sites/bernardmarr/2018/09/24/what-are-artificial-neural-networks-a-simple-explanation-for-absolutely-anyone/?sh=9d356ea12457. Accessed 21 Sept. 2023.

Pallavi Suyog Uttekar, MD. “What Is the Golden Hour in Medicine?” *MedicineNet*, MedicineNet, 26 Feb. 2021, www.medicinenet.com/what_is_the_golden_hour_in_medicine/article.htm. Accessed 21 Sept. 2023.

“Pandas Introduction.” *W3schools.com*, 2022, www.w3schools.com/python/pandas/pandas_intro.asp#:~:text=What%20is%20Pandas%3F,by%20Wes%20McKinney%20in%202008. Accessed 30 Sept. 2023.

Ray, Sunil. “Learn How to Use Support Vector Machines (SVM) for Data Science.” *Analytics Vidhya*, 12 Sept. 2017, www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. Accessed 19 Sept. 2023.

“Sklearn.metrics.accuracy_score.” *Scikit-Learn*, 2023, scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html. Accessed 27 Sept. 2023.

“What Is a .CSV File and What Does It Mean for My Ecommerce Business?” *BigCommerce*, 2023, www.bigcommerce.com/ecommerce-answers/what-csv-file-and-what-does-it-mean-my-ecommerce-business/. Accessed 24 Sept. 2023.

“What Is Multi-Class Classification | Deepchecks.” *Deepchecks*, 5 Aug. 2021, deepchecks.com/glossary/multi-class-classification/. Accessed 18 Sept. 2023.