**Phase-3**

**AI Based Diabetes Prediction**

|  |  |
| --- | --- |
| Date | 10-10-2023 |
| Team ID | Proj-212176-Team-2 |
| Project Name | AI Based Diabetes Prediction. |
| Maximum Mark |  |

**Introduction for Model Development and Model Evaluation:**

 The paper provides an overview of the current state of AI-based diabetes prediction and management. It also highlights the challenges faced in developing and implementing AI-based diabetes prediction models. The authors suggest that the predictive performance of AI will soon be maximized by a large amount of organized data and abundant computational resources, which will contribute to a dramatic improvement in the accuracy of disease prediction models for diabetes.

 Presents a study on the development and evaluation of machine learning models for type 2 diabetes prediction. The authors used three real-life diabetes datasets and nine feature selection algorithms for the evaluation. They compared the accuracy, F-measure, and execution time for model building and validation of the algorithms under study on diabetic and non-diabetic individuals. The performance analysis of the models is elaborated in the article.

 The paper lists the most popular AI models, including linear regression, deep neural networks, logistic regression, decision trees, linear discriminant analysis, naive Bayes, support vector machines, learning vector quantization, and k-nearest neighbors. Provides an overview of the top AI development frameworks and libraries in 2022. The paper highlights the most popular Python libraries for AI development, including TensorFlow, Keras, PyTorch, Scikit-learn, and Pandas.

**Coding for Model Evaluation:-**

**Program:-**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

df = pd.read\_csv('C:/Users/91638/Documents/diabetes.csv')

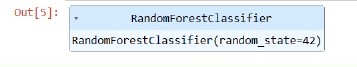
X = df.drop('Outcome', axis=1)

y = df['Outcome']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)



y\_pred = model.predict(X\_test)

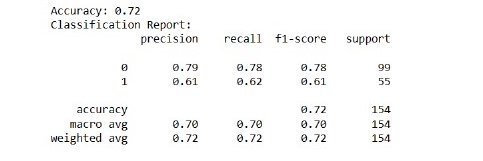
accuracy = accuracy\_score(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred)

print(f"Accuracy: {accuracy:.2f}")

print("Classification Report:")

print(report)



**Model Development:**

Developing an AI-based diabetes prediction module involves collecting and analyzing relevant data, selecting appropriate features, training the model using machine learning algorithms, and evaluating its performance. It's an exciting area of research with the potential to improve healthcare outcomes. Let me know if you need any specific information or assistance with your project.

**Model Development module:**

Next, you select appropriate features that are most informative for diabetes prediction. This could include factors like age, BMI, blood pressure, and glucose levels. After that, you split the dataset into training and testing sets.

Now comes the exciting part – training the model! You can use various machine learning algorithms like logistic regression, decision trees, or neural networks to train the model on the training data. The model learns patterns and relationships from the data during this phase.

Once the model is trained, you evaluate its performance using evaluation metrics like accuracy, precision, recall, and F1 score. This helps you assess how well the model is predicting diabetes.

**Coding for Model Evaluation:**

**Program:**

import numpy as np

from sklearn.datasets import load\_iris

from sklearn.tree import DecisionTreeClassifier

import matplotlib.pyplot as plt

from sklearn import tree

from sklearn import metrics

from sklearn.metrics import classification\_report

from sklearn.metrics import \*

from sklearn.metrics import mean\_squared\_error

iris = load\_iris()

X, y = iris.data, iris.target

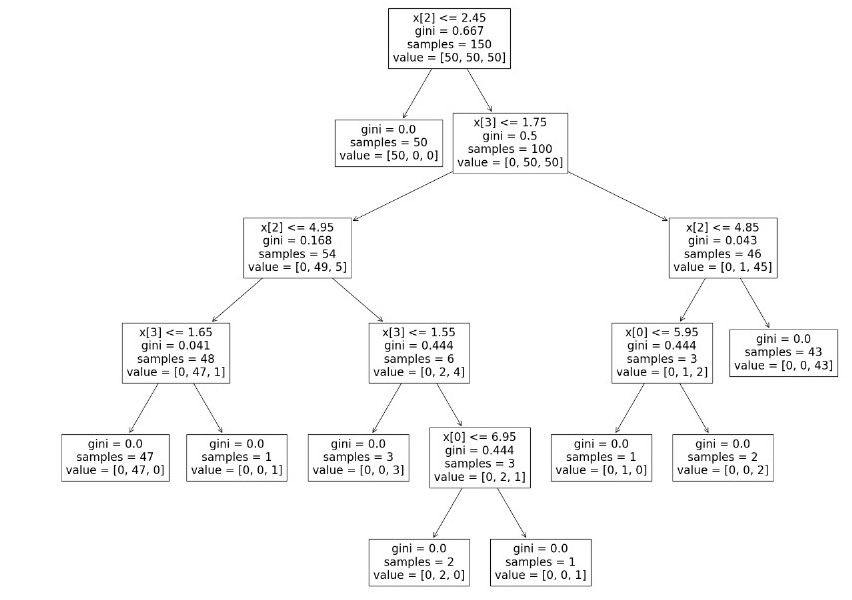
clf = tree.DecisionTreeClassifier()

clf = clf.fit(X, y)

plt.figure(figsize=(25,20))

tree.plot\_tree(clf)

plt.show()



print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))

score=r2\_score(y\_test,y\_pred)

print("The value of R squared is ",score)

print("The MSE is=",mean\_squared\_error(y\_test,y\_pred))

print("The RMSE value is = " , np .sqrt (mean \_squared \_error (y\_test ,y\_pred)))

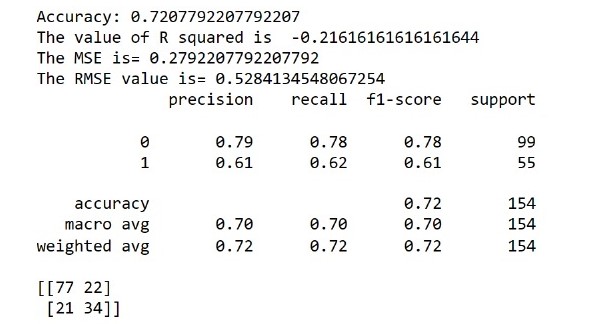
score1=score

report=classification\_report(y\_test,y\_pred)

print(report)

mat=confusion\_matrix(y\_test, y\_pred)

print(mat)



**Model Evaluation:**

To evaluate the performance of an AI-based diabetes prediction model, you can use metrics such as accuracy, precision, recall, and F1 score. These metrics help assess how well the model predicts diabetes based on the given data. Cross-validation and confusion matrices are commonly used techniques for evaluating the model's effectiveness. Let me know if you need more details or assistance with model evaluation.

**Model Evaluation Module:**

Model evaluation in AI-based diabetes prediction! Once you have trained your model, it's important to evaluate its performance. This helps you understand how well the model is predicting diabetes based on the given data.

There are several evaluation metrics you can use, such as accuracy, precision, recall, and F1 score. Accuracy measures the overall correctness of the model's predictions. Precision tells you how many of the predicted positive cases are actually true positives. Recall, on the other hand, indicates how many of the true positive cases were correctly identified by the model. F1 score is a balance between precision and recall.

To evaluate the model, you can use techniques like cross-validation and confusion matrices. Cross-validation helps assess the model's performance on different subsets of the data, while confusion matrices provide insights into the model's predictive power. It's important to note that model evaluation is crucial to ensure the reliability and accuracy of your predictions.

When it comes to model development in AI-based diabetes prediction, there are various approaches you can take. Some common types include logistic regression, decision trees, random forests, support vector machines, and neural networks. Each model has its own strengths and weaknesses, so it's important to choose the one that best suits your specific needs.

For model evaluation, you can use metrics like accuracy, precision, recall, and F1 score to assess the performance of your model. Cross-validation, where the data is split into multiple subsets for training and testing, helps validate the model's effectiveness. Additionally, techniques like ROC curves and confusion matrices provide insights into the model's predictive power. Remember, it's essential to choose the right model and evaluate it properly to ensure accurate and reliable predictions.

**Modularity:**

Modularity is a concept that is important in many fields, including artificial intelligence. In the context of AI, modularity refers to the ability to break down a complex system into smaller, more manageable parts or modules. This can help to reduce complexity and make it easier to understand and work with the system. [Modularity is typically expressed in terms of a hierarchical decomposition, where different components of a system are divided into separate functional units](https://www.cs.ubc.ca/~mack/Publications/PCAR06.pdf).

**Reusability:**

In the context of AI, reusability refers to the ability to reuse pre-built AI solutions and components, and customize them without coding. [This will allow AI solutions to be created without requiring scarce AI talent or costly IT resources](https://www.infoworld.com/article/3644968/how-no-code-reusable-ai-will-bridge-the-ai-divide.html). In academia, reusability of tools or components allows for wider scientific impact. [Contemporary software solutions are increasingly based on “Artificial Intelligence” (AI) models, and it is tempting to explore how much of the AI-based tools can be reusable and applied in a different context.](https://www.infoworld.com/article/3644968/how-no-code-reusable-ai-will-bridge-the-ai-divide.html)

**Encapsulation:**

Encapsulation is a concept used in object-oriented programming to bundle data and methods into easy-to-use units. To better understand encapsulation, view it as a medicine capsule that can’t viewed from the outside. Similarly, in the realm of programming, encapsulation involves bundling data variables and the methods that manipulate the data into a single private unit, like a capsule. It conceals the inner workings and exposes only what is necessary. Encapsulation is important because it provides a powerful way to store, hide, and manipulate data while giving you increased control over it. Encapsulation can be used when dealing with secure data or methods because it can restrict which functions or users have access to certain information.

**Abstraction:**

In AI, abstraction is used to represent knowledge in a way that is more manageable for computers. By using abstraction, AI systems can more easily identify patterns and make predictions. Abstraction can also be used to hide details that are not relevant to the task at hand. [It is a widely used concept in artificial intelligence to manage the use of different levels of detail in a representation language or the ability to switch between levels while preserving important characteristics](https://www.aiforanyone.org/glossary/abstraction).

**Testing and Validation:**

In AI, testing and validation are essential steps to make a robust supervised learning model. During training, the model is trained on the data in the training set. Validation data is used to provide an unbiased evaluation of the model fit during hyperparameter tuning of the model. It infuses new data into the model that it hasn’t evaluated before, allowing data scientists to evaluate how well the model makes predictions based on the new data. In machine learning, verification is testing that your product meets the mathematical description, specifications, and requirements you have written.

**Collaboration:**

Collaboration in AI refers to the ability of humans and AI to work together to accomplish a shared goal. [Collaborative AI refers to a viewpoint that looks beyond an individual’s cognition to include goal-driven tactical, operational, and strategic interactions of individuals with others (including other non-human cognitive agents) in order to develop far superior collective intelligence through computational modeling/evaluations of such interactions/engagements among the collaborating agents](https://researcher.watson.ibm.com/researcher/view_group.php?id=7806). To enable effective collaboration between humans and AI, organizations must facilitate a culture that embraces change, innovation, and continuous learning. [Developing AI literacy across all levels of your workforce, encouraging interdisciplinary collaboration, and creating an environment where humans and AI work hand in hand are essential steps](https://www.forbes.com/sites/forbesbusinesscouncil/2023/07/26/the-human-ai-symbiosis-embracing-collaboration-for-a-smarter-future/).