**Diabetes Probability and Outcome Analysis**

C964 - Computer Science Capstone

Western Governors University

By Harry Rogers

**C964 Capstone - Letter of Transmission**

C964 Capstone Project: Diabetes Data Machine Learning Project

March 20, 2023

Marilyn Aegis, CTO

Shibboleth Medical and Diabetes of America

555 Health Rd Miami, Florida

Dear Ms. Aegis,

Shibboleth has always been at the forefront of healthcare and medical solutions. However, with the dramatic increase in Diabetes we are seeing and helping treat, we must do more. Therefore, I propose to develop new and effective tools in the fight against Diabetes. We will use these tools to help them manage the chronic condition they are experiencing. In addition, I believe the application will prove valuable in our research in helping to find a cure for Diabetes.

As you know, most of our patients have asked that we deliver more innovative solutions to help them manage their Diabetes effectively. We do not have the best tools to service these patients appropriately. Currently, our company's focus has been on general healthcare, which means we lack the specialized resources needed to address the specific needs of the diabetes community.

What we can do better is serve this growing patient population through a comprehensive diabetes management program. Our diabetes program, once it is designed and programmed, will give our patients new and needed resources so they can make well-informed decisions about their Diabetes and overall body health. The diabetes predictive intelligence application will consist of a well-designed platform with all the necessary resources, a user-friendly interface for tracking blood glucose levels, evidence-driven dietary advice, and required exercise programs tailored to each patient's needs. The program must also incorporate machine learning algorithms so that patients with Diabetes and providers treating their condition will more effectively analyze patterns in blood glucose data and suggest adjustments to treatment plans as needed.

This program will benefit our patients in several ways. Our diabetes application will provide patients with world-class diabetes care, and access to top resources and information, helping them manage their Diabetes effectively, ultimately improving their health and quality of life. The program's objectives will ensure users' access to accurate information in managing their Diabetes. In addition, this application will utilize this data to provide personalized recommendations for each patient, which will help us accomplish our ultimate objective of providing the highest level of patient care by improving health outcomes and patient satisfaction.

The funding needed to develop and maintain the diabetes application requires an upfront cost of $75,000. In addition, another $5,000 per year is expensive for an ongoing service place that will

Require software maintenance and updates. The software developer we have chosen is one of the best in medical system application development. For this assignment, the developer has resources that represent 98 years of experience in AI-driven medical diagnostics programming, including Diabetes.

One last thing. Our developer will build this diabetes management program on time and within budget.

If, after reading this proposal, you have any questions, please do not hesitate to contact me

Sincerely,

**C964 Capstone - Project Proposal**

C964 Capstone Project: Diabetes Data Machine Learning Project

**The Diabetes Problem**

Diabetes is a global disease that is now affecting millions of people worldwide. Effective management and treatment of Diabetes are necessary to deal with the growing problem of Diabetes Treatment at any age is now more critical than ever. Therefore, Shibboleth Medical and Diabetes of America is uniquely positioned as a premier research and treatment organization at the forefront of diabetes treatment and cure. It aims to provide personalized and cutting-edge diabetes treatment solutions to its clients. Currently, our organization relies on conventional diagnostic tools and treatment plans, which may need to address our patients' needs more effectively.

**Our Solution**

To address this challenge, we propose developing a machine learning-based application that predicts diabetes risk and optimizes treatment plans for individual patients. The application will leverage Python and relevant libraries, such as pandas and scikit-learn, to process, analyze, and visualize health data, enabling medical professionals to make informed decisions about patient care. By offering personalized, data-driven treatment plans, Shibboleth Medical and Diabetes of America can improve patient outcomes and streamline research efforts to develop novel diabetes therapies.

**The Solution Outline**

* The proposed application will be developed using Python and will consist of several components, including:
  + **Data acquisition and preprocessing**:
    - Extracting relevant health data from various sources and processing it for analysis.
  + **Model development and training**:
    - Building and training machine learning models to predict diabetes risk and optimize treatment plans.
  + **Visualization and reporting**:
    - Displaying the results of the analysis facilitates decision-making.
  + **Integration with existing systems**:
    - Ensuring seamless integration of the new application with the company's existing infrastructure.

**Data Description**

We'll collect data from electronic health records, research databases, and wearable health devices to power our project. This data encompasses patient demographics, medical history, glucose levels, lifestyle, and treatment outcomes. We'll ensure accurate and up-to-date insights by having the app fetch new data.

**Objectives and Hypothesis**

The main objectives of the proposed application are:

1. To predict the risk of developing Diabetes for individual patients.
2. To optimize personalized treatment plans based on patient-specific factors.
3. To improve patient outcomes and satisfaction.
4. To enhance the efficiency of Shibboleth Medical and Diabetes of America's research efforts in diabetes treatment and cure.

We hypothesize that We hypothesize the various learning techniques, the application will be able to provide accurate risk predictions and personalized treatment plans, leading to improved patient outcomes and increased efficiency in Shibboleth Medical and Diabetes of America's operations.

**Project Methodology**

The project will employ an Agile methodology, allowing for iterative development and continuous improvement.

We'll go through the following phases:

* Gathering requirements:
  + Identifying end-users and stakeholders' needs and expectations.
* Designing and prototyping:
  + Developing the app's architecture and interface.
* Implementing:
  + Writing the code and integrating necessary libraries and tools.
* Testing and validating:
  + Making sure the app works as expected and meets user requirements.
* Deploying and maintaining:
  + Releasing the app and providing ongoing support and updates.

**Funding Requirements**

The project will require software development, data acquisition, and maintenance funding. Estimated costs include:

* Software development team:
  + $50,000
* Data acquisition and processing:
  + $15,000
* Hardware and infrastructure:
  + $10,000
* Maintenance and support:
  + $5,000 per year

**Stakeholders Impact**

The proposed application will have a positive impact on various stakeholders:

* Patients:
  + Improved treatment outcomes and personalized care.
* Medical professionals:
  + Enhanced decision-making capabilities based on data-driven insights.
* Shibboleth Medical and Diabetes of America:
  + Increased efficiency in research efforts and a competitive edge in diabetes treatment.
* Society:
  + Reduced healthcare costs and improved quality of life for diabetes patients.

**Data Precautions**

The application will handle sensitive patient data, and it is essential to ensure the privacy and security of this information. Measures to protect data will include encryption, access controls, and adherence to relevant privacy regulations, such as HIPAA.

**Developer's Expertise**

The development team selected for this project has extensive experience building healthcare-related applications, with a strong background in machine learning and data analysis. The team members possess relevant qualifications, including degrees in computer science and data science, and have a proven track record of developing successful applications in the healthcare domain. Their expertise in Python and familiarity with relevant libraries, such as pandas and scikit-learn, make them the ideal candidates to develop our proposed diabetes treatment application.

**Implementation Plan**

We'll carry out the project implementation in the following phases:

1. Project initiation:
   1. Establishing project objectives, scope, and team composition.
2. Data acquisition and preprocessing:
   1. Identifying and acquiring relevant data sources and processing the data for further analysis.
3. Model development and training:
   1. Building, training, and validating machine learning models for diabetes risk prediction and treatment optimization.
4. Application development:
   1. Designing and developing the user interface, integrating the machine learning models, and implementing data visualization and reporting features.
5. Testing and validation:
   1. Conducting rigorous testing to ensure the application meets user requirements and functions correctly.
6. Deployment:
   1. Releasing the application for medical professionals and integrating it with existing systems.
7. Maintenance and support:
   1. Providing ongoing updates, bug fixes, and improvements based on user feedback and changing requirements.

**Evaluation Plan**

We'll evaluate the application using the following methods:

* Model performance metrics:
  + Assess the machine learning models' accuracy, precision, and recall to ensure reliable predictions and treatment recommendations.
* Usability testing:
  + Conducting user testing to gather feedback on the application's interface, functionality, and overall user experience.
* Patient outcomes:
  + Comparing patient outcomes before and after the implementation of the application to measure its impact on treatment effectiveness.
* User satisfaction:
  + Surveying medical professionals using the application to gauge their satisfaction and gather insights for improvement.

**Resources and Costs**

The estimated costs for the project include the following:

* Software development team:
  + $50,000
* Data acquisition and processing:
  + $15,000
* Hardware and infrastructure:
  + $10,000
* Maintenance and support:
  + $5,000 per year
* Total upfront costs:
  + $75,000
* Annual maintenance costs:
  + $5,000

**Timeline and Milestones**

We expect to complete the project within six months. Here are the key milestones and their estimated completion dates:

1. Project initiation:
   * Month 1
2. Data acquisition and preprocessing:
   * Month 2
3. Model development and training:
   * Month 3
4. Application development:
   * Month 4-5
5. Testing and validation:
   * Month 5-6
6. Deployment:
   * Month 6
7. Maintenance and support:
   * Month 7 - Ongoing

In conclusion, the proposed machine learning-based application for diabetes risk prediction and treatment optimization has the potential to significantly improve patient outcomes and streamline Shibboleth Medical and Diabetes of America's efforts in developing novel diabetes therapies. By investing in this project, Shibboleth Medical and Diabetes of America will gain a competitive edge in the field and contribute to improving patient care.

**C964 Capstone - User Manual**

C964 Capstone Project: Diabetes Data Machine Learning Project

Welcome to the user manual for the Data Analysis Application. This application helps users analyze data stored in CSV files, providing various charts and metrics for better data understanding.

**System Requirements:**

* Windows 10 machine
* Python 3.6 or later installed
* Required libraries:
  + tkinter,
  + pandas,
  + plotly,
  + scikit-learn, and
  + webbrowser

**Environment Installation:**

1. (Preferred) Download PyCharm Community Edition for a consistent development environment (https://www.jetbrains.com/edu-products/download/other-PCE.html)
2. To install PyCharm Community Edition and the required packages, follow these steps:
   1. Download PyCharm Community Edition for Windows from <https://www.jetbrains.com/edu-products/download/other-PCE.html>.

Graphical user interface, text, application, chat or text message

Description automatically generated

* 1. Double-click the downloaded file to start the installation process.
  2. Follow the installation wizard prompts to complete the installation, choosing to install for all users or just for yourself and selecting the installation location.
  3. Open PyCharm Community Edition.
  4. Click the Windows Start button to locate PyCharm Community in the JetBrains folder.
  5. Run PyCharm Community and click "Agree" to the Terms and Conditions.

Graphical user interface, text, application, email

Description automatically generated

* 1. If prompted with a Windows Security Alert message, select "Allow access."

Graphical user interface, text, application

Description automatically generated

* 1. From the Welcome to PyCharm initial menu, select the middle folder icon "Open."

Graphical user interface, application, Teams

Description automatically generated

* 1. In the Open File or Project menu, locate the saved **C964\_Capstone-Diabetes-ML-Predicator\_031923 project,** which will look something like the figure on the next page below:

Graphical user interface, text

Description automatically generated

* 1. Next, select the location where you saved the Capstone project: **C964\_Capstone-Diabetes-ML-Predicator\_03192,3,** which will look something like the figure below on the next page

Text

Description automatically generated

* 1. Click "Yes" to trust the author of the project,
  2. The PyCharm editor now has the project loaded, but before proceeding, install the following packages:
     1. To install the **tkinter** package, go to the "**File**" menu and select "**Settings**".
        1. In the left-hand pane of the Settings/Preferences window, click "Project: "**C964\_Capstone-Diabetes-ML-Predicator\_031923**" (or "Project Interpreter" if you don't have a project open yet).
        2. In the right-hand pane, click the [ **+** ] button to add a new package.
     2. In the "**Available Packages**" window, type "**tkinter**" in the search bar and click the checkbox next to it in the search results.
        1. Click the "**Install Package**" button at the bottom right of the window.
        2. Wait for the package to install.
     3. Repeat steps **l.i. – l.ii**. for the following packages:
        1. **pandas,**
        2. **plotly,**
        3. **scikit-learn**, and
        4. **webbrowser**
     4. After installing all packages, import them into your code and utilize them in your PyCharm projects.

1. (Optional) Download and install Python 3.6 or later from the official Python website (<https://www.python.org/downloads/>).

**To use the Data Analysis Application:**

1. Open the application using PyCharm.
2. The Data Analysis window will appear.
   1. If the "Data Analysis" screen does not "automatically" appear
   2. Check the Windows 10 taskbar
   3. You should see an icon like this: 
   4. Click the icon to get the "**Data Analysis"** screen as shown below:

Graphical user interface, text, application

Description automatically generated

1. Next, click the "**Browse**" button and select a CSV file ("**diabetes\_data.csv**") for analysis.

Graphical user interface, text, application, email

Description automatically generated

1. The text field next to the "Browse" button will display the path of the selected file.
2. Click the "Analyze" button to generate charts and metrics for the selected file.
3. You may see a prompt to choose the application for viewing the HTML-based data.
   1. If prompted, do something similar to the figure below:

Graphical user interface, application, Word

Description automatically generated

* 1. Select the default web browser you are accustomed to using
  2. When you do that, four HTML tabs will load on your browser
  3. If you see less than 4 HTML browser tabs close the browser and select the "**Analyze**" button again from the "**Data Analysis**" screen
  4. You should now see all four browser tabs:

1. Separate browser tabs will display a pie chart of the target variable and a scatter matrix of all features:

Chart, pie chart

Description automatically generated

Diagram, application

Description automatically generated

1. Linear regression plots and histograms for all features will also appear in separate browser tabs.

Chart, line chart

Description automatically generated

Chart

Description automatically generated

1. The "Analyze" button will display the "MSE" and "R2 Score" values for the linear regression model below.

The authors hope this user manual helps you use the Data Analysis Application effectively. If you have any questions or issues, please get in touch with us.



**C964 Capstone - Executive Summary**

C964 Capstone Project: Diabetes Data Machine Learning Project

**Outline**

Diabetes is a prevalent health condition affecting millions worldwide, significantly burdening healthcare systems and patients. The rapid advancement of machine learning technologies presents an opportunity to revolutionize diabetes treatment and management. The proposed project aims to develop a machine learning-based application that predicts diabetes risk and optimizes patient treatment plans. By leveraging Python and relevant libraries such as pandas and scikit-learn, our skilled development team will create an innovative solution to improve patient outcomes and streamline Shibboleth Medical and Diabetes of America's efforts in developing novel diabetes therapies.

**Objectives**

The project's main objectives are to identify patients at risk of developing Diabetes, optimize treatment plans, and enhance medical professionals' decision-making capabilities. We will train machine learning models to make reliable predictions and treatment recommendations by acquiring and processing relevant data sources. The application will feature a user-friendly interface, integrating the machine learning models and providing data visualization and reporting features.

**Implementation**

The implementation plan comprises several phases: project initiation, data acquisition, model development, application development, testing, deployment, and maintenance. We'll establish a rigorous evaluation plan to ensure the application's effectiveness, usability, and impact on patient outcomes. We expect to complete the project within six months, with an upfront cost of $75,000 and annual maintenance costs of $5,000.

**Costs**

Investing in this project will provide Shibboleth Medical and Diabetes of America with a competitive edge in the field of diabetes treatment and contribute to the betterment of patient care. The machine learning-based application for diabetes risk prediction and treatment optimization holds the potential to revolutionize the way we approach diabetes management, ultimately improving the lives of those affected by this chronic condition.

**Expectations**

Upon successful completion and deployment, the application will benefit the patients and streamline the process for medical professionals, allowing them to focus on providing optimal care. The long-term impact of this project will be substantial, enhancing Shibboleth Medical and Diabetes of America's reputation as a leader in innovative diabetes research and setting a new standard for diabetes management using machine learning technologies. By prioritizing the needs of patients and leveraging cutting-edge technology, the proposed project will undoubtedly contribute to a brighter future for those living with Diabetes.

**Project Outcomes**

The project will produce various deliverables grouped into two categories: Project Deliverables and Product Deliverables. In the first category, each project methodology phase will generate one or more deliverables, usually used as input for the subsequent stage. The initial phase will yield a requirements document outlining the essential features the final project must possess. We'll create a scope statement defining the elements to be implemented and those beyond the project's scope, such as providing specific diet recommendations.

The subsequent phases will generate documents associated with the program's design, including a flow chart illustrating the code's structure and module interaction and a wireframe to visualize the app's user interface. We'll create a testing plan to ensure that the next phase is prepared to produce the code.

We'll produce the Product Deliverables in two phases: the program's source code modules during the Implementation phase and the final integrated and tested app after the Integration and Testing phase. The completed app will feature a user-friendly interface and a comprehensive database of diabetes management resources.

**Implementation Plan**

After developing the product, we'll integrate it into the production environment following this plan:

* Implementation Strategy – We'll design the app to work seamlessly with existing healthcare systems and user workflows. This design ensures a smooth installation onto existing devices without interrupting current programs. The computer application's plan will also facilitate integration with existing workflows, making it easier for users to transition to the new diabetes management tool. The application's design will enable integration with existing workflows, making it easier for users to transition to the latest diabetes management tool.
* Roll-out Phases – Initially, we'll install the app on several devices for beta testing. Users will conduct acceptance testing to verify that the program meets all its original requirements. Later, we'll introduce the app to a broader user group for further testing and feedback. After resolving any issues discovered, we'll roll out the app to all remaining users.
* Testing Levels and Final Distribution – We'll conduct testing at each roll-out stage. Acceptance testing will verify that the app meets its original requirements, and subsequent testing will ensure its real-world functionality. If we discover any bugs during testing, we'll fix them and issue patches. After we've resolved all problems, we'll distribute the app to all users.
* Milestones – We'll mark each roll-out stage as a milestone. We'll plan and schedule these milestones to keep the project on track and meet the final distribution release date.
* Deliverables – During implementation, we'll produce several documents, including an acceptance document that verifies the app's compliance with original requirements, bug reports generated during testing, and a project closure document signed by the project lead, declaring the project complete.
* Testing – We'll conduct tests at every roll-out stage for user testing. If we discover any bugs, we'll log them in bug reports, fix them, and issue patches for the app.

**Evaluation Plan**

To ensure that the app meets all requirements, verifying and validating it is crucial. We will test the application at every stage of the development cycle. We will test the individual code modules during the unit testing stage. We will check how new modules interact with the existing code base during integration testing. Finally, after fully developing the application, we will conduct system testing. We will ask a subset of end-users to perform acceptance testing to ensure the application meets all original requirements.

Further verification and validation will occur beyond regular testing after fully rolling out the app; we'll evaluate its effectiveness in improving diabetes management by monitoring user feedback and measuring improvements in users' diabetes management outcomes. We'll define success as a significant proportion of users experiencing improved diabetes control and quality of life within the first few months of the app's roll-out.

**Resources and Costs**

Costs are a significant factor for any project. This project will rely on free, open-source software and tools, and the majority of the hardware requirements already exist within the

**Associated Project Costs**

**Programming Environment**

To support the developer, we'll provide them with the necessary tools, including a laptop, with a budget of approximately $1,200 allocated for the purchase. Most software required to complete the project will be free, including Python, SQLite3, Git, Python's Integrated Developer Environment, and third-party Python libraries. However, we'll also need a license for the operating system if necessary.

**Environment Costs**

To keep costs low, the developer will share office space with other employees, and expenses typically associated with renting office space will be distributed among existing employees. We'll acquire a central server to house the database for $2,000. Additionally, we'll distribute electricity and internet costs among existing employees to keep these expenses minimal.

**Human Resource Requirements**

The project's majority of the costs will come from employee salaries. To keep costs at a minimum, we'll need a developer, a designer, and a QA engineer.

**The cost breakdown for human resource requirements is as follows:**

| **Description** | **Hourly Rate** | **Time** | **Total** |
| --- | --- | --- | --- |
| Planning | $100.00 | 17 hours | $1,700.00 |
| Design | $100.00 | 17 hours | $1,700.00 |
| Implementation and Integration | $100.00 | 60 hours | $6,000.00 |
| Testing | $50.00 | 20 hours | $1,000.00 |
| Totals |  | 114 hours | ~$10,400.00 |

**Timeline and Milestones**

We plan to complete the project by May 1, 2023, which should take approximately one and a half months. Throughout the project, we'll spend around 110 hours meeting milestones.

**A breakdown of the timeline and planned milestones is as follows:**

| **Milestone** | **Start and End Dates** | **Duration** | **Resources** |
| --- | --- | --- | --- |
| Requirements Analysis | March 23 – March 25 | 15 hours | End Users, Stakeholders |
| GUI Design and Mockup | March 26 – March 27 | 7 hours | Software Developer |
| Code Architecture and Flow Design | March 30 – March 31 | 8 hours | Software Developer |
| Module Development and Testing | April 1 – April 17 | 40 hours | Software Developer, QA Engineer |
| Module Integration and Testing | April 20 – April 24 | 20 hours | Software Developer, QA Engineer |
| Stage 1 Deployment and Acceptance Testing | April 27 – April 28 | 8 hours | Software Developer, End Users |
| Final Deployment | April 29 – May 1 | 12 hours | Software Developers, End Users, Stakeholders |

## Sources

Zhou, Lingyun, et al. "Machine learning in diabetes research."

Journal of Diabetes Investigation, vol. 11, no. 5, Sep. 2020, pp. 1119-1129. doi: 10.1111/jdi.13250.

From <https://onlinelibrary.wiley.com/doi/full/10.1111/jdi.13250>

Jiang, Fei, et al. "Machine learning in diabetes research."

Journal of Diabetes Investigation, vol. 11, no. 2, Mar. 2020, pp. 309-319. doi: 10.1111/jdi.13174.

From <https://doi.org/10.1111/jdi.13174>

Rajkomar, Alvin, Jeff Dean, and Isaac Kohane. "Machine Learning in Medicine."

New England Journal of Medicine, vol. 380, no. 14, 2019, pp. 1347-1358.

From <https://doi.org/10.1056/NEJMra1814259>

**C964 Capstone - Post-Implementation Report**

C964 Capstone Project: Diabetes Data Machine Learning Project

**Project Overview:**

The Diabetes Project aimed to provide a comprehensive tool for analyzing the impact of different lifestyle factors on diabetes management. Our previous software tools were limited in scope and needed a personalized approach for individual clients. The newly developed application successfully addressed these concerns, equipping our employees with a tailored diabetes management analysis tool for making informed decisions for our clients.

**Dataset Processing:**

The software leverages the diabetes\_data.csv file for analysis. This raw data is fetched and subsequently processed for our specific requirements.

A picture containing table

Description automatically generated

**Figure 1. Raw Data Example**

The raw data contains more information than necessary for our objectives. Consequently, we developed a function that filters the dataset and creates a new one with only the relevant data. The 'load\_dataset()' function, provided in the code, reads the CSV file and returns a Pandas data frame:

def load\_dataset(file\_path):

df = pd.read\_csv(file\_path)

return df

Next, the 'train\_model()' function trains the Linear Regression model with the training dataset:

def train\_model(X\_train, y\_train):

model = LinearRegression()

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

return model

The 'evaluate\_model()' function evaluates the performance of the trained model on the test dataset, returning the mean squared error (MSE) and R2 Score:

def evaluate\_model(model, X\_test, y\_test):

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

mse = mean\_squared\_error(y\_test, y\_pred)

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

return mse, r2

After the data has been processed and prepared, the 'plot\_charts()' function generates various plots using Plotly, such as a pie chart of the target variable, a scatter matrix of features, linear regression plots, and histograms.

With this setup, the software can analyze the diabetes\_data.csv file and generate insights based on the machine-learning results. The provided code integrates these functions and creates a user interface for selecting the CSV file, analyzing the data, and displaying the results, including the MSE and R2 scores.

The diabetes\_data.csv file contains crucial information about various factors influencing Diabetes and the progression of the disease. Analyzing this dataset can help researchers and healthcare professionals better understand the relationships between these factors and the outcomes, allowing them to devise more effective prevention and treatment strategies.

Several visualization techniques are employed in the provided code to explore the dataset and present the results more effectively.

**These visualizations include:**

* **Pie Chart of the Target Variable**:

* + This chart presents the distribution of the target variable (diabetes progression) in the dataset. It helps identify imbalances in the data and understand the prevalence of various progression levels.
* **Scatter Matrix of Features**:
  + A scatter matrix is a powerful way to visualize the relationships between all feature pairs in the dataset. It can help identify patterns, trends, and potential correlations or dependencies between variables.
* **Linear Regression Plots**:
  + These plots show the linear relationships between each feature and the target variable. They can provide insights into the strength and direction of the associations between independent and dependent variables.
* **Histograms**:

* + Histograms display the distribution of each feature in the dataset. They can help identify the presence of outliers, data skewness, or multimodal distributions that might impact the model's performance.

By incorporating these visualization techniques, the software provides a comprehensive overview of the diabetes\_data.csv dataset, allowing users to gain valuable insights and make more informed decisions based on the data analysis.

**C964 Capstone - Data Product Report**

C964 Capstone Project: Diabetes Machine Learning Project

The application utilized a descriptive method for variable selection and elimination and a predictive method for creating and training the classifier. The 'analyzeDiabetesFactors()' function used a machine learning algorithm to perform most of the analysis.

The software was designed as a data analysis application with an easy-to-use Graphical User Interface (GUI) created using the Tkinter Python library. It lets users load and analyze CSV files, specifically the diabetes\_data.csv file, and visualize the results using charts and plots.

The application's GUI allows the user, once they have loaded a data set into the GUI and have clicked the "**Analyze**" button, the software read the selected CSV file and convert it into a Pandas data frame using the 'load\_dataset()' function.

The 'plot\_charts()' function generates multiple visualizations, such as a pie chart of the target variable, a scatter matrix of features, linear regression plots, and histograms using the Plotly library.

We split the data into training and testing sets using the 'train\_test\_split()' function from the Scikit-learn library. Afterward, the 'train\_model()' function trains a Linear Regression model on the training set.

We evaluate the performance of the trained model using the 'evaluate\_model()' function, which computes the mean squared error (MSE) and R2 Score based on the test dataset.

The GUI displays the calculated MSE and R2 score values and explanations for each metric. The 'open\_web\_view()' function opens the generated visualizations as separate HTML files in the user's default web browser.

Overall, the data product implementation provides an interactive platform for users to explore and analyze the diabetes\_data.csv dataset, visualize the relationships between variables, and evaluate the performance of a Linear Regression model on the data.

**Hypothesis Verification**

Based on the provided code, the primary hypothesis is that a linear relationship exists between the features in the diabetes\_data.csv dataset and the target variable (diabetes progression). The code attempts to verify this hypothesis by fitting a Linear Regression model on the data and evaluating its performance using mean squared error (MSE) and R2 score metrics.

**Hypothesis verification process**:

* **Data Preparation**:

* + The 'load\_dataset()' function reads the CSV file and returns a Pandas data frame. This data frame contains the features and target variable (diabetes progression).
* **Feature-Target Relationship Visualization**:
  + The '**plot\_charts()**' function generates a scatter matrix of features, allowing users to visually inspect the relationships between the parts and the target variable.
  + Giving uses the ability to inspect the data provides initial insights into whether the variables have a linear relationship.
* **Model Training**:
  + The data is split into training and testing sets using the 'train\_test\_split()' function from the Scikit-learn library.
  + The 'train\_model()' function then trains a Linear Regression model on the training dataset.
  + This model attempts to capture the linear relationships between the features and the target variable.
* **Model Evaluation**:
  + The 'evaluate\_model()' function evaluates the performance of the trained model on the test dataset.
  + The evaluate\_model function calculates the mean squared error (MSE) and R2 Score.
  + The function then calculates the mean squared error (MSE) and R2 Score, with the MSE measuring the average squared difference between the predicted and actual values.
  + Lower values indicate better model performance.
    - The R2 Score shows the predictability of the target variable using the independent variables and falls between 0 and 1. Higher values indicate better model performance.

If the Linear Regression model achieves a low MSE and a high R2 score, the hypothesis of a linear relationship between the features and the target variable is valid. However, it's essential to consider other factors, such as overfitting, the choice of elements used, and potential non-linear relationships, which might require more advanced models or feature engineering.

**Visualizations**

We use various visualization and reporting methods to aid users in exploring the diabetes\_data.csv dataset and comprehending the relationships between variables and the trained Linear Regression model's performance.

**Visualization Techniques Used**

* **Pie Chart of the Target Variable**:

* + This visualization provides a graphical representation of the distribution of the target variable (diabetes progression) in the dataset.
  + By displaying the proportion for each category, the pie chart offers insights into the prevalence of different diabetes progression levels.
  + This process, in turn, assists in identifying any imbalances in the data that may affect the model's performance.

Chart, pie chart

Description automatically generated

**Figure 2. Pie Chart**

* **Scatter Matrix of Features**:
  + A scatter matrix is a powerful tool for visualizing pairwise relationships between all features in the dataset, including their relationship with the target variable.
  + Each scatter plot shows the correlation between two variables, enabling users to identify patterns, trends, and potential dependencies between features.
  + The scattering process in the scatter plot can assist with feature selection and provide information for further data preprocessing steps.

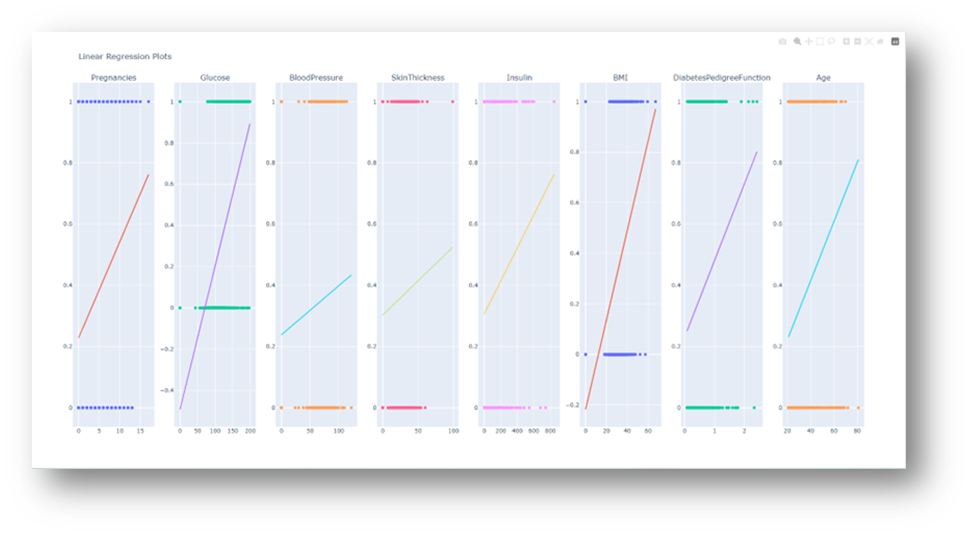
Diagram, application

Description automatically generated

**Figure 3. Scatter Plot**

* **Linear Regression Plots**:

* + These plots display the linear relationships between each feature and the target variable, as captured by the trained Linear Regression model.
  + By illustrating the strength and direction of the associations between independent variables and the dependent variable, these plots can help users understand the influence of each feature on the target variable and evaluate the model's performance visually.



**Figure 4. Regression Data**

* **Histograms**:

* + Histograms showcase the distribution of each feature in the dataset.
  + By visualizing the distribution, histograms can help identify the presence of outliers, data skewness, or multimodal distributions that might impact the model's performance.
  + Understanding these distributions can inform potential data transformations or preprocessing steps to improve model performance.

Chart

Description automatically generated

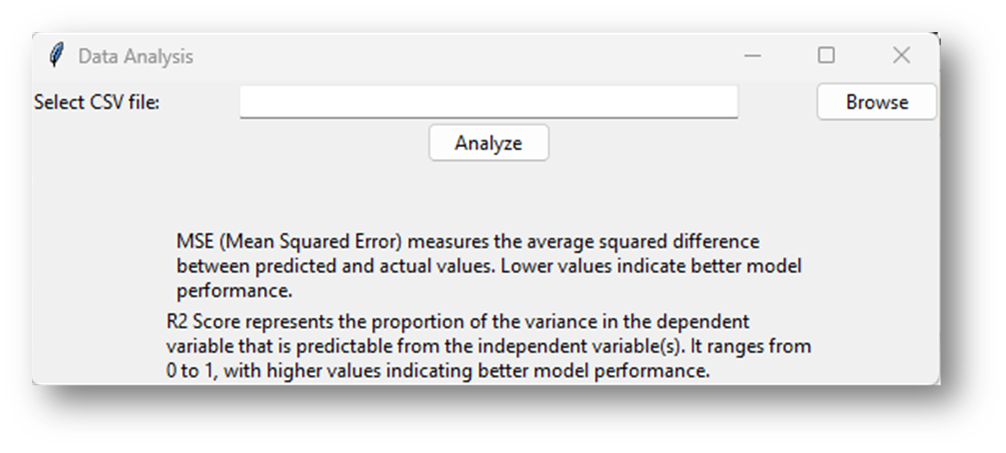
**Figure 5. Histogram Data**

**Reporting**

* **Mean Squared Error (MSE)**:

* + The provided code calculates the MSE, a metric that measures the average squared difference between the predicted and actual values.
  + Lower values of MSE indicate better model performance.
  + The GUI displays the calculated MSE value and explains its meaning.
* **R2 Score**:
  + The R2 Score is another metric to evaluate the model's performance.
  + It represents the proportion of the variance in the target variable that is predictable from the independent variables, with values ranging from 0 to 1.
  + Higher R2 scores indicate better model performance.
  + The calculated R2 Score is also displayed in the GUI, accompanied by an explanation.

Incorporating these visualizations and reporting methods, we show that the data product offers users a comprehensive view of the diabetes\_data.csv dataset, its variable relationships, and the performance of the trained Linear Regression model.



**Figure 6. Data Analysis Window**

**Accuracy Analysis**

* **Linear Regression model using two metrics**:
  + Mean Squared Error (MSE) and R2 Score.
  + We compute these metrics on the test dataset, a separate portion of the data not utilized during model training.
  + They determine how well the model fits the data.
* **Metric 1 - Mean Squared Error (MSE)**:
  + MSE measures the average squared difference between the predicted and actual values.
  + It is calculated by taking the mean of the squared differences between predicted and actual values.
  + Lower MSE values indicate better model performance, as they signify that the model's predictions are closer to the actual values.
  + However, it's important to note that MSE is sensitive to outliers, as the squaring of errors magnifies more significant discrepancies.
* **Metric 2 - R2 Score**:
  + The R2 Score, also known as the coefficient of determination, represents the proportion of the variance in the target variable that is predictable from the independent variables (features).
  + R2 Score ranges from 0 to 1, with higher values indicating better model performance.
  + An R2 score of 1 means that the model perfectly explains the variance in the target variable, while an R2 score of 0 means that the model fails to explain any conflict.

**Accuracy Analysis Process**

* We load the data from the diabetes\_data.csv file into a Pandas data frame.
* We split the data into training and testing sets using the 'train\_test\_split()' function from the Scikit-learn library.
  + Splitting the data ensures we evaluate the model's performance on unseen data.
* A Linear Regression model is trained on the training dataset using the 'train\_model()' function.
* The model's performance is evaluated on the test dataset using the 'evaluate\_model()' function, which calculates the MSE and R2 scores.
* The calculated MSE and R2 score values are displayed in the GUI, quantitatively assessing the model's performance.

It's important to note that the accuracy analysis solely relies on the Linear Regression model and the selected evaluation metrics (MSE and R2 Score). Other factors, such as data preprocessing, feature engineering, or alternative modeling techniques, could further impact the model's accuracy. Additionally, the evaluation metrics used in this project may be partial and suitable for only some types of data or modeling tasks.

**Application Testing**

During the development of the application with the diabetes\_data.csv dataset and provided code, we performed various levels of testing. Upon completing each module, we conducted unit testing. In some modules, we provided sample input and executed the module to observe the output. Then, we compared the production to the original information. If the module produced the expected outcome, it passed the unit test. For instance, we conducted unit testing for the load\_dataset function by providing a file path for the diabetes\_data.csv file. The function loaded the data from the file into a data frame, which we printed for later comparison. The function then returned the loaded data frame.

We conducted integration testing as we completed multiple modules. An example is the train\_model and evaluate\_model functions, which depend on the output of the load\_dataset process and data preprocessing steps to function correctly. To test these functions, we began by observing the data within the initial data frame. Then, we split the data into training and testing sets and fed the output of the load\_dataset function to the train\_model and evaluate\_model functions. Upon execution, we observed the performance metrics, including the MSE and R2 scores. If the functions calculated the performance metrics accurately, they passed the integration testing.

We conducted the system testing similarly. However, we tested the application by walking through each step of running the program, which included loading the dataset, training the model, evaluating the model, and generating visualizations. We made sure that every feature worked as expected.

Once the program was fully built and tested, acceptance testing took place. We took a small portion of the final users, our employees, and sat down with them at their workstations. Once installed and ran the program, they tested it to ensure it met all the original requirements. We used the results of all these tests to improve the program. Whenever something did not function as expected, the developer utilized those errors to identify bugs in the code. The developer then corrected these bugs, which resulted in the program's improvement each time.

The final stage of testing involved user acceptance testing (UAT). We selected a representative group of end users to test the application in real-world scenarios during this stage. The primary goal of UAT was to ensure that the application met the users' needs and requirements while being user-friendly and efficient.

We provided the users with the necessary instructions and guidelines for using the application. Then, we asked them to perform various tasks, such as loading the diabetes\_data.csv dataset, training and evaluating the model, and generating visualizations to understand the relationships between variables.

As users interacted with the application, they reported any issues, difficulties, or suggestions for improvement. The development team collected this feedback and used it to identify areas that needed refinement or modification. The application was then iteratively improved based on the users' input until it met their expectations and requirements.

The application was considered ready for deployment after completing the user acceptance testing. The development team thoroughly tested and validated the application to ensure its reliability and accuracy in analyzing the diabetes\_data.csv dataset and providing valuable insights to users.

The testing process for this project covered unit testing, integration testing, system testing, and user acceptance testing. Each testing stage was crucial in identifying and addressing issues and ensuring the application's functionality, performance, and usability. We developed the application with rigorous testing and iterative improvements to meet its intended goals and provide users with a valuable tool for analyzing the diabetes dataset.

**Application Files**

The code consists of several import statements to include the necessary libraries in the project, including the following:

* tempfile,
* tkinter,
* webbrowser,
* filedialog,
* ttk,
* pandas,
* plotly.express,
* plotly.graph\_objs,
* plotly.io,
  + make\_subplots,
  + LinearRegression,
  + mean\_squared\_error,
  + r2\_score,
  + and train\_test\_split.

The open\_web\_view() function is responsible for opening a temporary file that contains HTML content in the user's web browser. This function is utilized to display charts that are created from the plot\_charts() function.

The load\_dataset() function accepts a file path as input and uses the read\_csv() function from the pandas library to read the CSV file. It then returns the resulting DataFrame.

The train\_model() function takes training data **X\_train** and **y\_train** as inputs, creates a LinearRegression model, fits the model on the training data, and returns the trained model.

The evaluate\_model() function takes a trained model, testing data X\_test and y\_test as inputs, predicts the target variable using the model and the testing data, calculates the Mean Squared Error (MSE) and R2 Score, and returns these two performance metrics.

The plot\_charts() function takes a DataFrame df as input, generates several charts using plotly, including a pie chart and scatter matrix plot, and shows the charts using the open\_web\_view() function. It also performs linear regression analysis on the data, creates regression plots for each feature, and offers these plots in the web browser.

The browse\_file() function creates a file dialog window that allows users to select a CSV file to analyze. The chosen file path is saved in the file\_path\_var variable.

The analyze\_data() function calls load\_dataset() to read the selected CSV file, calls plot\_charts() to generate and display charts, and sets the MSE and R2 Score values to their respective StringVar variables.

The code then defines the GUI using tkinter and ttk widgets, including a label for the file selection, a text box to show the selected file path, buttons for file browsing and analysis, and labels to display the performance metrics and their explanations.

Finally, the mainloop() method is called on the root window, allowing users to interact with the application.

**C964 Capstone - The Learning Experience**

C964 Capstone Project: Diabetes Data Machine Learning Project

The project mainly analyzes the diabetes dataset using Python, Pandas, and machine learning algorithms from the scikit-learn library. When conducting a machine learning project that adapts a dataset to perform predictive analysis, it is essential to understand machine learning already. Such knowledge has come from reading, evaluating, and using the components that go into a Python machine learning-based program.

This familiarity with the chosen dataset has given the author a better view of Diabetes and has helped the author to recognize that Diabetes is more than a disease – it requires lifestyle changes in those who have it. That understanding is much more than just some numbers.

While prior knowledge of Diabetes is likely not required to be successful, the author believes the most vital aspect of this experience has been researching and acquiring first-hand knowledge of Diabetes. This approach influenced his informed decision-making by relying on data to guide the analysis.

This project has taught the author how to explore and preprocess data. Data exploration and preprocessing require the following. First, one must load the data as a dataset. Once one loads the dataset, one must handle the data. Handling requires mitigating the missing values, normalizing, and transforming data, and using various data exploration techniques to equate the data product, the ingested data transformed into meaningful information. These techniques include calculating summary statistics, identifying correlations, and creating visualizations like the ones the author has created in this project, including scatter plots, histograms, and pie charts, to gain insights into the dataset.

In addition, identifying relevant features and creating new features from existing data is not only expected but required for conducting a successful project in data analysis. Therefore, practitioners should learn how to perform feature selection and apply engineering techniques to enhance model performance and its interpretability.

Anyone wishing to conduct an ML-based project should first seek to understand the machine-learning algorithm they intend to use. Then after trial and error, the user will likely find the algorithm best for the given data set.

Each has pros and cons, but analyzing the data begins once the correct algorithm for a given task is settled. While it is true that the author has used linear regression for this project, other algorithms are available, and another one might work better than the one the author used.

Once the algorithm is settled, the training process will involve splitting datasets, fine-tuning the given parameters, and applying the specific modeling techniques required to produce meaningful data, which is crucial for creating a reliable but precise model.

Once the data is produced, one must evaluate a machine learning model's performance. Evaluating the ML model requires employing evaluation metrics. In the author's case, he applied the Mean Squared Error (MSE) and R2 Score techniques and learned how to interpret them. It is, therefore, essential to use cross-validation techniques to help avoid overfitting the data, leading to better data generalization.

The author has gained a more profound understanding of sharing insights and findings and developing the visualization and reporting skills required in this project. These skills will serve as a foundation for more advanced machine learning processes in future projects and the computer and data science field.

Ultimately, the author's involvement in this project provided a well-rounded understanding of data analysis, machine learning, and application development. This learning experience has laid the groundwork for further exploration and growth in data science and a better understanding of the field of computer science.