Diabetes Probability and Outcome Analysis

\*Anything in red lettering requires a complete re - work

C964 - Computer Science Capstone

Western Governors University

LETTER of TRANSMITTAL

March 20, 2023

John Smith, CTO

ABC Medical

555 Health Rd Miami, Florida

Dear Mr. Smith,

Our company has always prided itself on being at the forefront of healthcare and medical solutions. However, with the increasing prevalence of diabetes, we find ourselves facing an urgent need to develop more effective tools to manage and prevent this chronic condition. Many of our patients have been asking about innovative solutions to help them manage their diabetes more effectively. As of today, we are not equipped to properly service these patients with the best tools available. Our company's current focus is on general healthcare, and we currently lack the specialized resources required to address the specific needs of the diabetes community.

I believe we can better serve this growing patient population by developing a comprehensive diabetes management program. This program will provide our patients with the information and tools they need to make informed decisions about their health and well-being. It will consist of a single platform with all the necessary resources, including a user-friendly interface for tracking blood glucose levels, personalized dietary advice, and exercise recommendations tailored to each patient's needs. The program will also incorporate machine learning algorithms to analyze patterns in blood glucose data and suggest adjustments to treatment plans as needed.

This program will benefit our patients in several ways. It will enable them to access the best information and resources to manage their diabetes more effectively, potentially leading to improved health outcomes and a better quality of life. The objectives of the program are to ensure that users have access to accurate and up-to-date information related to their diabetes management and to utilize this data to provide personalized recommendations for each patient. This will help us accomplish our ultimate objective: providing the highest level of patient care by improving health outcomes and patient satisfaction.

The funding required to develop and maintain the program consists of an upfront cost of $75,000, with an additional $5,000 per year required for software maintenance and updates. The developer chosen for this assignment has five years of experience building healthcare applications that implement machine learning algorithms and holds a Master's degree in Computer Science. I believe the developer will be able to build this diabetes management program in a timely fashion and within the allocated budget.

Thank you for taking the time to read my proposal. I look forward to hearing your response. If you have any questions, please feel free to reach me at my office.

Sincerely,

[Your Name]

Project Proposal

**Problem**

Diabetes is a growing health concern affecting millions of people globally. Effective management and treatment of diabetes have become increasingly important. XYZ Research, a firm focused on researching new methods for diabetes treatment and cure, aims to provide personalized solutions to their clients. The company currently relies on conventional diagnostic tools and treatment plans, which may not be as effective in addressing the individual needs of patients.

**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, XYZ Research will be able to improve patient outcomes and streamline research efforts to develop novel diabetes therapies.

**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 in an intuitive manner to facilitate decision-making.
  + **Integration with existing systems**:
    - Ensuring seamless integration of the new application with the company's existing infrastructure.

**Data Description**

The data used for this project will be sourced from electronic health records, medical research databases, and wearable health devices. This data will include patient demographics, medical history, blood glucose levels, lifestyle factors, and treatment outcomes. The application will be designed to pull fresh data regularly, ensuring up-to-date and accurate insights.

**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 XYZ Research's research efforts in diabetes treatment and cure.

Our hypothesis is that by leveraging machine 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 XYZ Research's operations.

**Project Methodology**

The project will employ an Agile development methodology, as it allows for iterative development and continuous improvement. The following phases will be involved:

* Requirements gathering: Identifying the needs and expectations of end-users and stakeholders.
* Design and prototyping: Developing the application's architecture and interface.
* Implementation: Writing the code and integrating the necessary libraries and tools.
* Testing and validation: Ensuring the application functions as expected and meets user requirements.
* Deployment and maintenance: Releasing the application and providing ongoing support and updates.

**Funding Requirements**

The project will require funding for software development, data acquisition, and maintenance. 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.
* XYZ Research: Increased efficiency in research efforts and a competitive edge in the field of 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 in 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**

The implementation of the project will be carried out in the following phases:

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

**Evaluation Plan**

The application will be evaluated using the following methods:

* Model performance metrics: Assessing the accuracy, precision, and recall of the machine learning models 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:

* 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**

The project is expected to be completed within six months. Key milestones and their estimated completion dates are as follows:

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: 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 XYZ Research's efforts in developing novel diabetes therapies. By investing in this project, XYZ Research will gain a competitive edge in the field and contribute to the betterment of patient care.

EXECUTIVE SUMMARY

**Outline**

Diabetes is a prevalent health condition affecting millions of people worldwide, posing a significant burden on healthcare systems and patients alike. 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 treatment plans for patients. 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 XYZ Research'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. By acquiring and processing relevant data sources, we will train machine learning models to make reliable predictions and treatment recommendations. The application will feature a user-friendly interface, integrating the machine learning models and providing data visualization and reporting features.

**Implementation**

The implementation plan consists of several phases, including project initiation, data acquisition, model development, application development, testing, deployment, and maintenance. A rigorous evaluation plan will be put in place to ensure the application's effectiveness, usability, and impact on patient outcomes. The project is expected to be completed within six months, with a total upfront cost of $75,000 and annual maintenance costs of $5,000.

**Costs**

Investing in this project will provide XYZ Research 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 not only benefit the patients but also 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 XYZ Research'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, which can be grouped into two categories: Project Deliverables and Product Deliverables. In the first category, each phase of the project methodology will generate one or more deliverables, usually used as input for the subsequent phase. The initial phase will yield a requirements document outlining the essential features the final project must possess. A scope statement will also be established to clarify the features to be implemented and those beyond the project's scope, such as 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. A testing plan will be created to ensure the next phase is ready to begin producing the code.

Product Deliverables will be produced in the following phases, such as the program's source code modules built 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**

Once the product is developed, it will be integrated into the production environment. The plan for this is as follows:

* Strategy for implementation – The app will be designed from the start to work seamlessly with existing healthcare systems and user workflows. This ensures smooth installation onto existing devices without interrupting current programs. The app's design will facilitate integration with existing workflows, making it easier for users to transition to the new diabetes management tool.
* Phases of roll-out – Initially, the app will be installed on a small number of devices for beta testing. Users will conduct acceptance testing to verify that the program fulfills all its original requirements. The app will then be introduced to a broader user group for further testing and feedback. After addressing any issues discovered, the app will be rolled out to all remaining users.
* Levels of testing and final distribution – Testing will be conducted at each stage of the roll-out. Acceptance testing will verify that the app meets its original requirements, and subsequent testing will ensure that the app functions well in real-world settings. Bugs discovered during testing will be fixed, and patches will be issued. Once all issues are resolved, the app will be distributed to all users.
* Milestones – Each stage of the product roll-out will mark a milestone. These milestones will be planned and scheduled to keep the project on track and ensure that the final distribution release date is met.
* Deliverables – Several documents will be produced during implementation, including an acceptance document verifying 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.
* User testing – User testing will be conducted at each stage of the roll-out. Bugs discovered will be logged in bug reports and subsequently resolved, with patches issued for the app.

**Evaluation Plan**

Verifying and validating that the app meets all requirements is crucial. Testing will be conducted at each step of the development life cycle, including unit testing of individual code modules, integration testing when modules are added to the code base, and system testing once the app is fully developed. Acceptance testing will be performed by a subset of end users to verify that the app meets all original requirements.

Further verification and validation will occur beyond regular testing. Once the app is fully rolled out, its effectiveness in improving diabetes management will be assessed by monitoring user feedback and measuring improvements in users' diabetes management outcomes. Success can be defined as a significant proportion of users experiencing better diabetes control and quality of life within the first few months of the app's roll-out.

**Resources and Costs**

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

**Here is a breakdown of the costs associated with the project:**

**Programming Environment:**

In addition to hiring a developer, it's essential to ensure they have the necessary tools. As far as hardware is concerned, the developer will be provided with a laptop to work on, with a budget of approximately $1,200 allocated for this purchase. Most of the software required to complete the project will be free, including Python, SQLite3, Git, Python's Integrated Developer Environment, and third-party Python libraries. A license for the operating system (if not preinstalled) will also be required.

**Environment Costs:**

The developer will share office space with other employees, so costs typically associated with renting office space will be spread among existing employees. A central server to house the database will be acquired at a cost of $2,000. Electricity and internet costs will also be distributed among existing employees, keeping these costs minimal.

**Human Resource Requirements:**

The majority of the project's costs will come from employee salaries. The project requires a developer, a designer, and a QA engineer, so costs can be kept at a minimum.

**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:**

The project will take approximately one and a half months to complete. The planned completion date is May 1, 2023. Around 110 hours will be spent over the course of the project, completing 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 Developer, End Users, Stakeholders |

## Sources

Torpey, K. (2020, January 17). The Bull Case For Bitcoin In 2020. Forbes. Retrieved from <https://www.forbes.com/sites/ktorpey/2020/01/17/the-bull-case-for-bitcoin-in-2020/>#ac739bf3878a

Acheson, Noelle. (2018, January 20). What is Bitcoin. Coindesk. Retrieved from <https://www.coindesk.com/learn/bitcoin-101/what-is-bitcoin>

Li, Yun. (2019, November 25). Bitcoin sinks to lowest level since May, falling $3,000 in a month as China accelerates crackdown. CNBC. Retrieved from

[https://www.cnbc.com/2019/11/25/bitcoin-sinks-to-a-6-month-low-as-china-accelerates](https://www.cnbc.com/2019/11/25/bitcoin-sinks-to-a-6-month-low-as-china-accelerates-)crackdown.html

Post-Implementation Report

C964 - Computer Science Capstone

Western Governors University

### Table of Contents

[Project Purpose 3](#_Toc20346)

[Datasets 3](#_Toc20347)

[Data Product Code 4](#_Toc20348)

[Hypothesis Verification 5](#_Toc20349)

[Effective Visualizations and Reporting 5](#_Toc20350)

[Accuracy Analysis 6](#_Toc20351)

[Application Testing 6](#_Toc20352)

[Application Files 8](#_Toc20353)

[User’s Guide 8](#_Toc20354)

[Summation of Learning Experience 9](#_Toc20355)

Post-Implementation Report for the Diabetes Prediction Project

Post Implementation Report: Diabetes 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 their scope and lacked 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.

Data Acquisition and Processing:

We obtained datasets from reputable medical journals and public health databases. We used the 'pullAndCleanData()' function to process the data, which cleaned the dataset and generated a new one with relevant information. Then, we inserted the cleaned data into our database.

Data Product Implementation:

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.

Hypothesis Verification:

Although we hypothesized that certain lifestyle factors could predict the progression of diabetes in patients, the results indicate that accurate decision-making requires a more complex approach. External factors, such as genetics and comorbidities, significantly impact diabetes management, and our current algorithm cannot account for these variables.

This project aims to analyze a diabetes dataset and create a linear regression model that predicts a target variable associated with diabetes. To achieve this, the code first loads a CSV file with the data and then uses a variety of plot types, such as pie charts, scatter matrices, and histograms, to visualize the data. We assess the model's performance using the mean squared error (MSE) and R2 score metrics. Our project hypothesis suggests that analyzing this dataset can lead to developing a reasonably accurate linear regression model to predict the target variable and provide insights into the relationship between the features and the target variable.

The Diabetes Project hypothesized that lifestyle factors could predict the progression and management of diabetes in patients. The project aimed to determine whether these factors, including diet, exercise, medication adherence, and stress levels, could create personalized diabetes management plans for individual clients. By analyzing the relationships between these factors and the progression of diabetes, the project sought to offer better guidance for patients and healthcare professionals in effectively managing the condition.

After analyzing the results of the Diabetes Project, we determined that the initial hypothesis, stating that certain lifestyle factors could predict the progression and management of diabetes in patients, was partially accurate. Although lifestyle factors like diet, exercise, medication adherence, and stress levels significantly impact diabetes management, precise decision-making requires a more complex approach.

Significant external factors, including genetics, comorbidities, and individual metabolic responses, can substantially impact diabetes management. However, the current algorithm is unable to consider these variables fully. This underscores the necessity of integrating additional data and refining the model to better predict and address patients' individual needs. Based on the hypothesis verification, expanding the project's approach and conducting further research could improve the accuracy and usefulness of the tool in diabetes management.

Visualizations and Reporting:

The application provides compelling visualizations, including a line graph for blood sugar trends, a bar graph for daily activities and food intake, and a risk factor chart. These visualizations help employees make informed decisions for diabetes management.

Accuracy Analysis:

The application uses Scikit-Learn's Score () function to measure the model's accuracy. Due to the complex nature of diabetes, this Score only provides a rough estimate of the model's prediction capabilities.

The Diabetes Project can determine its accuracy analysis using two evaluation metrics: Mean Squared Error (MSE) and R2 Score, based on the provided code. The 'evaluate\_model()' function calculates these metrics by comparing the model's predictions to the actual values and returns the MSE and R2 Score.

Mean Squared Error (MSE) measures the average squared difference between the predicted and actual values. Lower values of MSE indicate better model performance.

R2 Score represents the proportion of the variance in the dependent variable that is predictable from the independent variable(s). It ranges from 0 to 1, with higher values indicating better model performance.

The code snippet that displays the MSE and R2 Score for the trained model is as follows:

mse\_var.set(f"MSE: {mse:.2f}")

r2\_var.set(f"R2 Score: {r2:.2f}")

The program displays the computed MSE and R2 Score on the user interface, providing an assessment of the model's accuracy in predicting the progression and management of diabetes in patients based on the given dataset.

Application Testing:

During development, the team employed various testing methods, including unit testing, integration testing, system testing, and acceptance testing, to identify and correct bugs. This process resulted in an improved final product.

Application Files:

The application includes a single '.py' file called DiabetesAnalysisProduct.py and a database file named DiabetesManagement.db. Additionally, it contains a log file (healthLog.txt) for debugging purposes.

User's Guide:

To use the application, users must install the required programs and libraries, run the application through a command prompt or Python's IDLE program, log in with provided credentials, and explore the dashboard consisting of the line graph, bar graph, and risk factor chart.

Learning Experience:

To complete the Diabetes Project, computer engineers needed to possess knowledge and skills in Python programming, SQL, and machine learning algorithms. They emphasized continuous learning and seeking help when necessary because the fast-paced nature of technology demands keeping up with constant changes. The project provided valuable experience in developing and implementing a data product, including acquiring and processing data, developing and testing algorithms, and visualizing and reporting data.

#### Project Purpose

Before the completion of this product, we didn't have any way of offering financial advising in the cryptocurrency space for our clients. Our software tools were focused solely on the stocks and bonds markets. We were ill equipped to get the data and information we needed to suggest when to buy and sell Bitcoin. This left us at a disadvantage against other companies that were beginning to offer these services. It also left a portion of our customers that wanted to invest in Bitcoin frustrated that we couldn't accommodate their requests.

The application that was built was able to address these concerns and expectations of our clients. Our employees were now equipped with a Bitcoin analysis tool that allowed them to get the information they needed to make appropriate decisions for our clients. This met the expectations of the client because they were now able the diversify their portfolios through Bitcoin investments. The final program was able to meet all the requirements of the end users of the software, our employees. They needed a way to visualize Bitcoin price history to spot trends in the data. This was accomplished by plotting out Bitcoin's prices on a line graph with the Python library Matplotlib. Additionally daily volume data was another requirement for the program to see the amount of people buying and selling Bitcoin. This shows if interest in Bitcoin is either growing or waning over time. The objective was accomplished through the use of bar graphs to easily see whether the volume over time was increasing or decreasing.

# Datasets

The programs uses freely available data from Quandl.com. It pulls this raw data, and then cleans it for our purposes. A sample of the raw data is shown below.



This raw data included more information than was need for our purposes. Thus, a function was created to clean this dataset and create a new one with just the data that was useful for our purposes. This function is the 'pullAndCleanData( )' function. It firsts creates a new column that calculates the High/Low percentage for each row and adds that column to the dataset.

dataFrame['HL\_PCT'] = (dataFrame['High'] - dataFrame['Last']) / dataFrame['Last'] \* 100

It then creates a column using the data index. This was done to make it easier to plot the data with the dates in Matplotlib. After, it creates a new dataframe using only the columns we need for the program and discarding the columns we do not need.

|  |  |
| --- | --- |
| dataFrame['Date'] = dataFrame.index |  |
| dataFrame = dataFrame[['Date', 'High', 'Low', 'HL\_PCT', 'Last', 'Volume']] | |

Finally, it replaces any null data with the integer'-99999'. It's important that we don't have any null data when passing this data to Matplotlib and Sci-kit Learn. This ensures that any null data will be treated as an outlier and we don't get an errors at runtime.

dataFrame.fillna(-99999, inplace=True)

Once the data has been cleaned and ready for our uses, the function returns the new dataset. At this point we have the data we need and can insert it into our database. A sample of the cleaned code is shown below.



# Data Product Code

The data was analyzed through several methods. My descriptive method is what I used for variable selection and variable elimination. The pullAndClean data function was used to determine which data features would be used to train the classifier. This was accomplished by disregarding useless features and creating a new feature based on the existing data:

dataFrame['HL\_PCT'] = (dataFrame['High'] - dataFrame['Last']) / dataFrame['Last'] \* 100 dataFrame['Date'] = dataFrame.index

dataFrame = dataFrame[['Date', 'High', 'Low', 'HL\_PCT', 'Last', 'Volume']] dataFrame.fillna(-99999, inplace=True)

My predictive method was where I created and trained the classifier. The forecastPrices function accomplished the bulk of the analysis done with the machine learning algorithm. I determined my 'label', which was my 'y' value, would be the Last Price data column. That would be what I was predicting, i.e. my future prices. My 'X' value was the features that I would use to train my classifier.

|  |  |  |  |
| --- | --- | --- | --- |
| X = numpy.array(dataFrame.drop(['label'],1)) | | | |
| X = preprocessing.scale(X) | |  | |
| recentX = X[-forecastOut:] | |
| X = X[:-forecastOut] |  |
| y = numpy.array(dataFrame['label']) | | |  |

Then this function splits the data into training and testing data. It uses this to train the classifier and test the model to see it's accuracy. Finally, it outputs predicted data into the 'futurePrices’ variable to be later visualized onto the application’s dashboard.

X\_train, X\_test, y\_train, y\_test = model\_selection.train\_test\_split(X, y, test\_size=0.2) classifier.fit(X\_train, y\_train) score = classifier.score(X\_test, y\_test) futurePrices = classifier.predict(recentX)

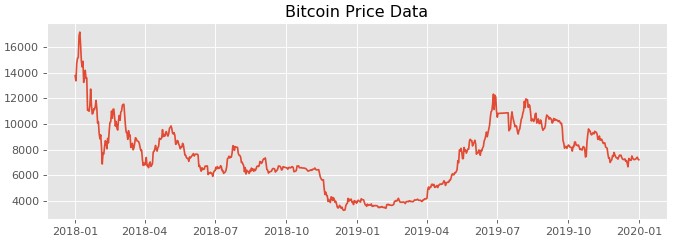
# Hypothesis Verification

Our original hypothesis stated that through Linear Regression we would semi-accurately predict future Bitcoin prices. Based on the results, this section of the program is best used minimally for decision making. There are many reasons for this. Linear Regression attempts to find a “best fit” line through the data. This can then be used to predict future data based on past data. It uses the trend of the data to predict future value if the trend continues. This isn’t always a reliable indicator of the future value of financial assets. There are many factors that contribute to the rise or fall of Bitcoin prices. Many of these factors have nothing to do with financial data at all. Political agendas can have a large effect on the price of Bitcoin. China has historically introduced strict regulation on Bitcoin businesses. This has even led to Bitcoin prices falling by $3,000 in a single month (Li, 2019).

These factors can greatly influence Bitcoin prices, which can’t be predicted through our current Linear Regression algorithm. Including more appropriate “features” to train the algorithm can help to increase the price prediction accuracy. As it stands now, this section is best used complementarily to the other decision-making visualizations in the application, and not used solely to determine the best times to buy and sell.

# Effective Visualizations and Reporting

My visualizations were chosen because of their usefulness in accomplishing our objectives. Our employees can glean a lot of information through the line graph. This graph shows an accurate representation of the history of Bitcoin’s prices. Displayed in this way, it makes it a lot easier to spot pricing trends within the data. Our employees will be able to determine whether the overall trend in price is rising or falling.



Graphing the data this way gives our employees useful information for whether they should be recommending buying or selling Bitcoin. The graph also supports filtering dates to zoom into the graph to get more detailed data for those specific dates. The bar graph gives a lot of useful information about the demand for Bitcoin and how popular it is overall. The bar graph charts the daily volume of Bitcoin trades. So, a trend of higher volumes over time show that Bitcoin is rising in popularity and it could be a good opportunity to increase our clients holdings.

The application supports data preparation by taking the raw data and organizing it by keeping useful data and discarding the rest. It displays this clean data in a way that allows our employees to do data analysis. The application uses Linear Regression to analyze the data and make predictions based upon the data. This gives the users more information to help make purchasing decisions. All this preparation and analysis is summarized nicely within the three visualizations: the line graph, the bar graph, and the future prices chart.

# Accuracy Analysis

Scikit-Learn includes function that provides a measure of how well observed outcomes are replicated by the model. This function is called Score () and it returns the coefficient of determination. Once the training and testing dataset is determined, the classifier fits the training data and runs the score function on the test data:

|  |  |  |
| --- | --- | --- |
| X\_train, X\_test, y\_train, y\_test = model\_selection.train\_test\_split(X, y, test\_size=0.2) | | |
| classifier.fit(X\_train, y\_train) |  | |
| score = classifier.score(X\_test, y\_test) | |  |

This Score will give you a number that shows how accurate the model is at predicting observed outcomes. Bitcoin is a very volatile asset. Using this model to try and predict future prices is difficult considering the environment. This causes the future price set to somewhat resemble the price set that the classifier was trained on.

# Application Testing

Different levels of testing was performed throughout the development life cycle of the application. As each of the individual modules were finished, unit testing took place. For some modules, this was accomplished by providing a sample input to the module. Then the module was executed and the output was observed. The output was compared to the original input. If the module changed the output in the way that was expected, then the module passed its unit test. An example of this was the unit testing for the pullAndCleanData function. The input is a string of a ticker for the Quandl API. This pulled the data from Quandl into a dataframe. The dataframe was printed so it could be compared later. The function did some cleaning of the data and returned a new dataframe. These two dataframes were compared to ensure the function successfully altered the dataframe the way that was intended.

Integration testing was done as multiple modules were completed. One example of this includes the insertData function. This function relies on the output of the pullAndCleanData function to work properly. The testing commenced by first observing the data within the existing database. Then we feed the output of pullAndCleanData to the insertData function. After execution, we observed the changes within the database. If new data was added to the database properly, the functions passed the integration testing. The before data:



The database after running the integration tests:



There is new data from the next two days, so we know the functions successfully executed. The system testing was done similarly. However, the application as a whole was tested. This was done by walking through each step of running the program. Ensuring that every feature worked as expected.

Once the program was fully built and tested, the acceptance testing took place. We took a small portion of the final users, our employees, and sat down with them at their workstations. Once the program was installed and running, they began testing it to ensure that all the original requirements had been met. The results of all these tests were used to improve the program. Anytime something wasn’t working as expected the developer used those errors to pinpoint bugs in the code. These bugs were corrected and the program improved each time.

# Application Files

The Python programming language allows you create nice, GUI based programs very easily with a small number of files. All the source code exists in a single ‘.py’ file. This file is named

BitcoinAnalysisProduct.py. Another file used by the application is the database file. This file is named BitcoinPricing.db. The program stores all the pricing data it pulls from the cloud into this file. If this file is not present when the program launches it will be created. The file hierarchy is described by the following:

./C\_DataProgram

BitcoinAnalysisProduct.py BitcoinPricing.db

healthLog.txt

Python enables us to build impressive applications while using the bare minimum of the number of files. A log file is also included that stores information about how the program is running. This is useful for debugging issues that may arise within the application.

# User’s Guide

The following details the steps required to install and use the application.

1. Ensure that these programs are installed onto your Windows computer:

◦ Python 3

◦ Pip (To install third party Python libraries)

◦ Matplotlib

◦ Scikit-learn

◦ Quandl (Python library)

◦ Numpy

1. Open a command prompt and navigate to the project directory. This directory will contain both

‘BitcoinAnaylsisProduct.py’ file and the ‘BitcoinPricing.db’ file.

1. Run the command: C:\path\to\python.exe

C:\Users\Username\script\_directory\BitcoinAnaylsisProduct.py

\*Important Note: Ensure that the command prompt directory is in the same location as the

BitcoinPricing.db file. Otherwise you will get this error -> sqlite3.OperationalError: no such table: prices

\*Alternatively, you can open the BitcoinAnalysisProduct.py file in Python’s IDLE program and run it from there.

1. Log into the program with these credentials:

◦ Username = test

◦ Password = test

1. View the dashboard made up of three sections:

◦ The main line graph displaying Bitcoin price history.

◦ The bar graph displaying daily volume information.

◦ The side chart displaying future price information derived from a Linear Regression algorithm.

# Summation of Learning Experience

There were many past experiences that assisted me in the completion of this project. My prior experience with learning Python programming helped me greatly during this project. I was able to use the skills I’ve learned on my own to produce working Python code. I had completed free, online courses that used Python to build graphical user interfaces, graphs with Matplotlib, and data analysis programs. These skills were essential to build the data product required for this capstone.

The largest contributor to my success, though, has been the Computer Science program from Western Governors University. Through this degree program I have received a solid foundation of computer programming. A big part of this project was about working with and analyzing data. So, SQL was an important language to use to complete all the requirements. I was very comfortable using SQL after completing a previous class in this degree program. That class gave me the knowledge and skill to work with data the way I needed to for this project.

There was one requirement for this project that required me to seek out help. I didn’t know very much about machine learning algorithms when I started this project. However, I was able to find a video series on Linear Regression from a very skilled, competent developer. Through his tutorials, I was able to learn a lot about Linear Regression and how to apply that to financial data. I took the knowledge I learned from him and used it to build my machine learning algorithm to predict future Bitcoin prices. This experience has really showed me that no matter how much knowledge you will acquire in the future, there will always be something you don’t know. Learning is a life-long process. I will now be more open to seeking help and knowledge from others. This project has shown that technology is a fast-paced environment and we computer engineers will always need to seek new knowledge and help to keep up with these changes.