**Dylan S. Franklin (Student ID: 001295234)**

**4/4/2023**

**Computer Science Capstone**

**Western Governors University**

**C951 Computer Science Capstone**

**TABLE OF CONTENTS**

[**LETTER OF TRANSMITTAL 2**](#_un1jiro8k300)

[**PROJECT RECOMMENDATION/PROPOSAL 3**](#_5u22ocvwe5ei)

[Problem Summary - 3](#_wewofmbhlav5)

[Application Benefits - 3](#_jzyw6eu3mciq)

[Application Description - 3](#_mgxr10wuafn5)

[Data Description - 4](#_lvfdcp69hdbx)

[Objective and Hypothesis - 4](#_4higm67sc0t6)

[Project Methodology - 5](#_2wkw3w6s924p)

[Funding Requirements - 5](#_6052hjq73jrw)

[Stakeholders Impact - 6](#_d6638zssnn5i)

[Data Precautions - 6](#_aq3nu4xd242j)

[Developer’s Expertise - 6](#_wdb4eobeqx7c)

[**EXECUTIVE SUMMARY 7**](#_67d1b9dtvixp)

[Problem Statement - 7](#_41l9s34hykky)

[Customer Summary - 7](#_t9siwlv9d3bw)

[Existing System Analysis - 7](#_7yi0esgo6jx7)

[Data - 8](#_cjo1ft5mkmu6)

[Project Methodology - 8](#_f7fe6a7ipr3t)

[Project Outcomes - 9](#_igitfep6iin6)

[Implementation Plan - 9](#_mz9oxvmymud3)

[Evaluation Plan - 10](#_8g33dme1sygf)

[Resources and Costs - 11](#_uj82qaymqozv)

[Timeline and Milestones - 12](#_slyo7tj2edoj)

[**POST IMPLEMENTATION REPORT 14**](#_9ukvbg415rgd)

[Project Purpose - 14](#_gl3gbjweo0wf)

[Datasets - 14](#_k3vjykwmfvx)

[Data Product Code - 16](#_ey5uf2jg6gi4)

[Hypothesis Verification - 17](#_lm1p4e574wnc)

[Effective Visualizations and Reporting - 17](#_49whhffoezmq)

[Accuracy Analysis - 19](#_xv795j9te9r0)

[Application Testing - 20](#_bjrno11xepjb)

[Application Files - 21](#_uuxx5s7keui8)

[User’s Guide - 22](#_3j66ku2yvwva)

[Summation of Learning Experience - 23](#_8q6nlxo7hnvt)

[**SOURCES 24**](#_399dmc20xc59)

# **LETTER OF TRANSMITTAL**

April 4, 2023

John Doe, CTO

777 Fairway Drive

Los Angeles, CA

Dear Mr. John,

Gold Gear has always been at the forefront of technological changes and how they can be used to support Gold Gears’ needs. The automotive industry is as competitive as ever and profits and losses are dictated by how accurately costs can be estimated during the early planning phases of automotive development. The faster and closer an estimate of cost can be made, the better.

Lately, Gold Gear has been looking to invest in new markets outside of the United States. As it stands today, I believe we’re not well enough equipped to invest in any new markets and suggest we create a prediction tool for vehicle cost first. This vehicle cost prediction tool would estimate the prices of vehicles based on their parts.

A prediction tool like this is extremely valuable to the company, as it allows for optimal pricing of their vehicles based on the vehicle's parts. This, in return, will allow the company to set competitive prices that are accurate and attractive to customers worldwide while keeping profitability high.

The total amount of money needed to create and maintain this prediction tool is $15,000 upfront, with an additional $1,000 a year for maintenance. This takes into account the collection and cleaning of data, the development of the prediction model and its features, and more. The developer assigned to this prediction tool is fairly experienced with machine learning, which will be a considerable aspect of this tool's creation. This developer also holds a Bachelor's degree in Computer Science, so his knowledge of automation, computation, and information is sure to aid him during this project's development as well. All things considered, I believe this developer will be able to provide a prediction tool that’ll meet requirements in a timely and costly manner.

Regardless of the decision, I thank you for taking the time to read my proposal. I look forward to hearing your thoughts on this suggestion. For any further details or questions feel free to contact me at [dfrank@goldgear.com](mailto:dfrank@goldgear.com) or visit me at my desk during work hours.

Sincerely,

Dylan Franklin

# **PROJECT RECOMMENDATION/PROPOSAL**

## **Problem Summary -**

Gold Gear has remained a United States car manufacturing company for decades but wishes to invest in new markets outside of the United States to increase its influence and profits. As it stands now, Gold Gear isn’t well enough equipped to invest in any new markets due to the lack of information regarding vehicle pricing outside of the United States.

The creation of a vehicle cost prediction model would allow Gold Gear to accurately price vehicles from across the entire world without needing to build the vehicle by merely using part information from past-built vehicles. A prediction model like this would allow Gold Gear to optimally price their vehicles based on the vehicle's parts. This would allow the company to set competitive prices that are accurate and attractive to customers worldwide while keeping profitability high. The finished product will be very valuable to Gold Gear, and make the transition to a wider market beyond the United States much smoother.

## **Application Benefits -**

The benefit of a vehicle cost prediction model is that it’ll allow Gold Gear to set competitive prices that are accurate to customers worldwide, not just in the United States. This prediction model will also give Gold Gear a competitive advantage against other automation companies by helping Gold Gear identify trends early and adjusting pricing as needed. The prediction model will also support better decision-making by providing Gold Gear with valuable information and data about the costs of individual vehicle parts, which will help the company make better-informed decisions about inventory management and pricing.

## **Application Description -**

The finished product will be a standalone application written in Python using many supported libraries and Jupyter Notebooks to help import and display data. The finished product will ALSO include a web application version of the Jupyter Notebook using Binder. Such libraries included in the development of the application include numpy, pandas, matplotlib, seaborn, scipy, ipywidgets, etc. Python was specifically the chosen language for developing this application due to libraries like these, as they are excellent for displaying data and data relationships, as well as supporting machine learning algorithms/models. Binder is a web application that will allow us to create an interactive and shareable environment using our Jupyter Notebook. Binder also can be thought of as a security feature, as nobody without the binder link can access the web application. Included with the application is a single CSV file that acts as all the data used to build the prediction model.

## **Data Description -**

The data used to create this application will be a comma-separated value (CSV) file from Kaggle that includes 205 rows and 26 columns of data. The following is a basic outline of SOME of the columns in the CSV file.

**ID** - A numeric (int64) column. Represents the PRIMARY KEY.

**symboling** - A numeric (int64) column.

**name** - A categorical (object) column. Represents the vehicle's name.

**fueltypes** - A categorical (object) column. Represents the fuel type used. (diesel or gas)

**aspiration** - A categorical (object) column. Represents the aspiration type used. (standard or turbo)

**doornumbers** - A categorical (object) column. Represents the number of doors.

**carbody** - A categorical (object) column. Represents the type of car body a vehicle has. (convertible, hatchback, sedan, etc.

… MANY MORE …

-**compressionratio** - A numeric (float64) column. Represents the volume of the cylinder and combustion chamber.

-**horsepower** - A numeric (int64) column. Represents the vehicle's horsepower.

-**price** - A numeric (int64) column. This is our target value or what we are predicting. Represents the vehicle's TOTAL price.

A limitation of the dataset that we will need to overcome is the number of categorical columns in the dataset. Since we plan to use a regression algorithm to predict our vehicle’s total price we need to convert these categorical columns into numeric columns for them to be used in the algorithm. We need to do this because regression algorithms primarily deal with predicting a numeric value (price) based on other numeric data, not categorical data.

## **Objective and Hypothesis -**

The objectives of this product/application are as follows.

* Use the data to make visualizations that are valuable to the company.
* Using the data, create a semi-accurate price prediction model that the user can use.
  + Make using the price prediction model a simple task.
* By doing all of the above, better prepare Gold Gear for its goal of reaching a wider market outside of the United States to increase influence, competition, and profits.

I hypothesize that by creating a price prediction model using vehicle parts through Linear Regression, reliable total vehicle price predictions can be made. This prediction of a vehicle’s total price, I believe, would allow Gold Gear to optimally price their vehicles based on the vehicle's parts when they do plan to expand their market to other countries outside of the United States. If this product is built and kept up to date, Gold Gears' influence, competition, chance of market growth, and profits would all increase.

## **Project Methodology -**

This project will use the Waterfall Methodology. The reason behind choosing the Waterfall methodology over something like Agile is that our requirements are well-defined and won’t change. This project is also fairly small and won’t need to go through a cycle of iterative development/testing like what you would do in an Agile environment. Because of this, using something simpler and most cost-effective like the Waterfall Methodology makes the most sense. The development will be done one step at a time with little to no going back to previous steps once development on a certain step is completed. The following is how the project will align with the waterfall methodologies steps…

* **Requirements:** During this phase, the objectives and requirements of the product will be discussed and gathered. A *project plan* will be created here to ensure no roadblocks like scope creep occur during the development of the product.
* **Design:** Hardware/software requirements must be established here. Plus, a general design of the product must be identified here using the previous steps' requirements and objectives (High/low-level design requirements).
* **Implementation:** Python code will be written here that meets the design requirements.
* **Testing:** The developer(s), will hand over the project to the software quality assurance testing team to look for bugs and errors that need to be fixed.
* **Deployment:** Once the project is running as intended with most/all bugs fixed, the completed project will be given to the end users for official business use.
* **Maintenance:** Unknown bugs may appear as time goes on from constant use of the project by the end-user. These bugs will be fixed as they come. Additional features may be added as future enhancements if needed/requested.

## **Funding Requirements -**

Most of this project will be created using free, open-source software and tools. Jupyter Notebooks, Binder, and all the libraries used are free to use. The total amount of money needed to create and maintain the project is stated in the letter of transmittal but will be discussed in depth here. The development of this project is estimated to take roughly 150 hours. Because the software development team for this project consists of one developer at a base pay of $100/hr, that leaves us with the project's estimated upfront cost of (100 x 150) $15,000. This takes into account the collection and cleaning of data, the development of the prediction model and its features, and more. In total, the amount of money needed to create and maintain the project is $15,000 upfront, with an additional $1,000 a year for maintenance.

## **Stakeholders Impact -**

The successful completion of this project would put Gold Gear in a very good spot to go through with its plan of expanding to new markets across the globe. This would affect stakeholders like…

* **Customers:** Many customers from outside of the United States would now have access to Gold Gears products and services. This may lead to an increase in competition between Gold Gear and other local automotive companies, which could potentially lower prices.
* **Shareholders:** Extremely likely to benefit from the company's market expansion as it would increase the company's profitability which could then lead to higher stock prices.
* **Employees:** By expanding a company's market, more job opportunities open up in new locations.

## **Data Precautions -**

The data used in this project was obtained for free on Kaggle. This means the data isn’t protected or sensitive and is fully compliant with standard privacy laws. However, in the case that we were using sensitive data, the following are some general guidelines we would need to follow…

* Use asymmetric/symmetric **encryption** like RSA, which is used for secure data transmissions.
* Implement **access control**, and apply the principle of least privilege to ensure unwanted access to information by those who don’t need it isn’t possible.
* **Comply with laws** regarding data privacy, gathering, storing, and removal. Such laws as HIPAA, GDPR, and more should be examined when handling data.
* **Properly dispose** of sensitive data.
* **Train employees** to recognize and handle sensitive data.

## **Developer’s Expertise -**

As stated in the letter of transmittal, the developer assigned to work on this project is fairly experienced with machine learning, which will be a huge aspect of this project’s creation. This developer also holds a Bachelor's degree in Computer Science, so his knowledge of automation, computation, and information is sure to aid him during this project's development. All things considered, this developer should be able to complete this project and all of its requirements in a timely and costly manner.

# **EXECUTIVE SUMMARY**

## **Problem Statement -**

Gold Gear has remained a United States car manufacturing company for decades but wishes to invest in new markets outside of the United States to increase its influence and profits. As it stands today, we are not well enough equipped to invest in any new markets. We need to create a prediction tool for vehicle cost first. This vehicle cost prediction tool would estimate the prices of vehicles based on their parts.

## **Customer Summary -**

One type of “customer” is those inside of the company that will be using the finished product/prediction tool to estimate the total price of vehicles all the time. The finished product will need to be installed for all of the employees that plan on using it. Employees that have a familiarity with data science should find navigating the product easy, as they’re likely to see many data visualization plots that they’ve seen before via other products. These data visualizations include distribution plots, heat maps, scatterplots, and more. That said, most if not all employees will already have the sufficient skillset to make use of the product via the prediction tool it provides through a machine learning algorithm, which is fairly easy to navigate and use.

Another type of “customer” are customers from outside of the United States that, because of the finished project, now have access to Gold Gears products and services. As it stands right now, Gold Gear isn’t ready to invest in new markets outside of the United States, but with this finished project, they’ll be more than ready. In this case, customers from outside of the United States can expect to receive Gold Gear automotive vehicles in due time.

## **Existing System Analysis -**

All of the tools needed for the project are available as free, open-source software, and are already being used. Python is the high-level programming language of choice due to its flexibility with many data visualization and machine learning libraries, all of which are also free and open-source. The libraries that will be used include numpy, pandas, matplotlib, seaborn, scipy, and ipywidgets.

## **Data -**

Data will be collected from Kaggle and downloaded as a CSV file. For the program to output meaningful information, this CSV file must be read into the program as a DataFrame which will be done using the pandas library. CSV is the chosen file format due to its simplicity and popularity with libraries like pandas. A small look at the structure of the CSV file can be seen under the “data-description” section of the project recommendation form. Because we’re dealing with many columns of data that range in type from numeric to categorical, we will need to do some basic cleaning of the data. Certain categorical columns will need to be transformed into numeric columns and irrelevant columns of data like the “ID” and “symboling” columns will be removed as they have no use to us. Any potential duplicate rows in the data will be removed as well.

## **Project Methodology -**

As discussed above, the project will follow the Waterfall Methodology. The project's requirements are well established, and in terms of size, the project is not that large. Because of this, doing iterative development is unnecessary, ruling out the effectiveness of using an Agile Methodology. Also, due to there only being one Software Engineer developing the project, a more sequential development path makes more sense, making the Waterfall Methodology the best option for the development of the project. The following are the steps of the Waterfall Methodology and how they will be executed in the project’s creation.

* **Requirements**
  + Establish requirements and objectives. (Requirements document)
  + Define scope via a project plan to avoid roadblocks like scope creep.
  + Project kick-off meeting to discuss the objectives and requirements and scope with users and stakeholders.
* **Design** 
  + Hardware and software requirements must be established. Documents from the previous step must be brought and analyzed here.
  + The general design of the product must be created based on the requirements.
* **Implementation**
  + Python code with the help of external libraries will be written to match the general design and meet the requirements and objectives from the previous steps.
  + Unit testing will occur here to ensure a small chunk of the project is functional before it’s implemented into the project.
* **Testing**
  + Developers will hand over the project to the software quality assurance testing team to look for bugs and errors that need to be fixed.
  + Integration and system testing will occur here to ensure no errors or bugs are present.
* **Deployment**
  + The completed project will be given to Gold Gears employees for official business use.
  + Acceptance testing is done to ensure the project meets requirements and is bug-free and functional.
* **Maintenance**
  + Unknown bugs may appear as time goes on from constant use of the project by the end-user. These bugs will be PATCHED as they come.
  + Additional features may be added as future enhancements if needed/requested.

## **Project Outcomes -**

The project's outcomes or deliverables include a fully functional Juypter Notebook and Web Application that visualizes the data from the CSV file and includes a price prediction model using a linear regression algorithm that the user uses. The process of creating the project will produce many deliverables as well. Plenty of documents such as the requirements document, the scope statement document, the budget document, etc are deliverables that are a result of finishing a specific step in the project's lifecycle.

## **Implementation Plan -**

**Strategy for Implementation**

The strategy for the implementation of this project is as follows. First, we must develop well-defined requirements, objectives, and goals. Next, we need to outline our project’s milestones and deadlines and assign resources to the project. Then, identify potential risks and roadblocks of the project through a risk management plan and monitor performance for these said risks and issues. Lastly, a plan for quality assurance should be made. An important step in this project’s case is ensuring the finished product will work with Gold Gears' current technology. To ensure this, the project will be built around Gold Gears' current hardware and workflows.

**Phases of Rollout**

When the project is finished, the product will be given to a small number of employees to use and test. If there are no issues the product will continue to slowly continue to be given to employees in waves each getting bigger until the majority of the company is using the product. If employees encounter issues with the product, the project will be returned to the developer to fix said issues, and the rollout process will repeat once more.

**Levels of Testing**

Testing will be done at all levels of the project’s rollout process. The first stage of testing will be primarily focused on acceptance testing and the product's ability to meet the planned objectives and requirements. Once passed, the product will then be tested for functionality by assigning a small number of employees to use the product and report any bugs and issues that arise. Depending on the results the product may either be returned to the developer to be further worked on or deployed to more employees. This process will continue a few times until the majority of the company is using the product.

**Dependencies and Milestones**

At the end of each level of testing a milestone will have been made. Meaning there will be a few milestones, all of which will be planned beforehand to know when a milestone has been met. With every milestone made, the product will get further to completion.

**Deliverables**

Many documents will be created in the process of the implementation of the project. Such documents include many status reports that focus on the overall project’s progress and any issues. Bug reports will be created during the rollout process. Any changes made to the project’s scope will be done through a change request document. Design documents, user documents, and technical documents are also deliverables created during the process of the implementation of the project. A project closure document will also be created when the project is completed and finished.

**User Testing**

User Testing will be done at all phases of the rollout process. All discovered bugs will be logged in the bug report by the user. From there, these bugs will be fixed by the developer.

## **Evaluation Plan -**

To ensure the finished application meets the requirements and objectives of the company, tests will be done at every waterfall step of the application's development lifecycle. Unit testing will be done on every function, module, method, and cell of code in the Jupyter Notebook to ensure functionality. When a cell of Python code works, integration testing will be done to check if it works with the entire program. Once the project is completed, acceptance testing will be done to check if the project meets the requirements and objectives discussed during the requirements phase of the project's life cycle.

We must also test the usability of the finished product to Gold Gear. The prediction model must function with enough accuracy to be useful to Gold Gear in their mission to expand their market outside of the United States. A benchmark of a “Root Mean Squared Value” (RMSE) value of $5000 or lower will be set on the product to ensure it’s reliable and useful to Gold Gear. This value indicates how far off on average the prediction model is, any value above $5,000 is too off of a prediction to be useful to Gold Gear and its mission. Meeting this benchmark by having an RMSE value of below $5,000 means the prediction model and thus the project is useful to Gold Gear and will help them in investing in prediction vehicle costs for new markets.

## **Resources and Costs -**

Most of the project utilizes free, open-source software during the implementation phase of the project’s life cycle. Hiring a developer for this project is where most of the costs come into play. That said, the following are the costs associated with the project.

**Programming Environment -**

The total cost of all software and hardware necessities needed in the creation of this project will be $0. This project will be created using Python 3.7 in Jupyter Notebooks using several third-party libraries like numpy, pandas, matplotlib, seaborn, etc, all of which are free to use and open-source. The developer working on the creation of this project will be using his laptop, so nothing is needed in terms of hardware from the company either.

**Environment Costs -**

The developer will be working on the project remotely so no office space or renting space is required. Electricity and hardware are the responsibility of the developer. The total cost of providing an environment to the developer will be $0.

**Human Resource Requirements -**

This is where most of the cost of the project will come into play. Paying the single developer his salaried income will cost what is most of the project's budget. The cost breakdown is as follows.

| **Project Step** | **Hourly Rate** | **Time** | **Total** |
| --- | --- | --- | --- |
| Requirements gathering/Planning | $100.00 | 25 Hours | $2,500.00 |
| Design | $100.00 | 20 Hours | $2,500.00 |
| Implementation | $100.00 | 70 Hours | $7,000.00 |
| Testing | $100.00 | 20 Hours | $2,000.00 |
| Deployment | $100.00 | 15 Hours | $1,000.00 |
| **TOTAL** | — | 150 Hours | $15,000.00 |

Additionally, maintenance on the project will cost $1,000 a year.

## **Timeline and Milestones -**

The following is a projected timeline that the project will follow along with important Milestones. The project in total will last 2 months.

**MILESTONE 1- PROJECT PLANNING.**

RESOURCES NEEDED: Project Managers, Stakeholders (End Users)

(25 HOURS) (MARCH 6 - MARCH 10)

This milestone will include the gathering of requirements and the definition of the scope and goals of the project.

**MILESTONE 2- DATA COLLECTION, CLEANING, AND VISUALIZATION.**

(20 HOURS) (MARCH 13th - MARCH 15th)

RESOURCES NEEDED: Data Scientist/Data Analyst

This milestone will focus on the collection and cleaning of relevant data. Statistical analysis will be performed on the data to identify relationships within the data to be visualized. This information and its insights will be valuable to both the creation of the rest of the project and Gold Gear.

**MILESTONE 3- MODEL DEVELOPMENT**

(70 HOURS) (MARCH 16th - MARCH 27th)

RESOURCES NEEDED: Data Scientist/Software Engineer/ML Specialist

This milestone will entail all development of the project. The linear regression model will be created, and the data collected during the previous milestone will be split into training and testing datasets for the prediction model’s use. The model may be tuned to increase its performance and accuracy.

**MILESTONE 4- MODEL EVALUATION.**

(20 HOURS) (MARCH 28th - MARCH 31st)

RESOURCES NEEDED: Data Scientist/Software Engineer/ML Specialist

This milestone will track the model's performance by using the Root Mean Squared Value (RMSE). This metric will tell us, on average, how off the prediction model’s guesses were for the training and testing datasets provided during the previous milestone. Depending on the results, the model may need to be fine-tuned.

**MILESTONE 5- EARLY DEPLOYMENT**

(8 HOURS) (APRIL 3rd - APRIL 4th)

RESOURCES NEEDED: SQA Engineer, End Users, ML Specialist

This milestone includes the project's rollout process. The model will be integrated into Gold Gears' environment slowly, starting with a few people and growing exponentially if no bugs or issues are found. Monitoring and testing the project's performance is crucial here.

**MILESTONE 6- FINAL DEPLOYMENT**

(10 HOURS) (APRIL 5th - APRIL 7th)

RESOURCES NEEDED: SQA Engineers, End Users, ML Specialist

The model is officially deployed and integrated into Gold Gears environment. Performance is still monitored and maintenance is done when needed.

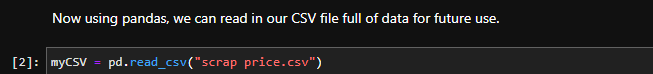
Milestones must be done in sequential order. 1 -> 2 -> 3 -> 4 -> 5 -> 6.

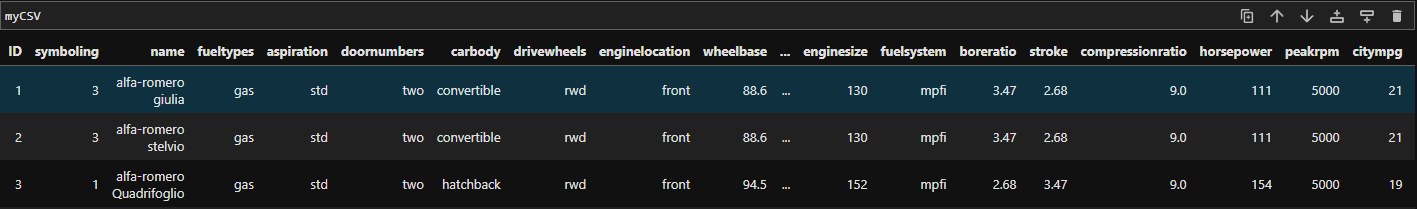
# **POST IMPLEMENTATION REPORT**

## **Project Purpose -**

As a reminder, the purpose of this project was to provide Gold Gear with a prediction model that would allow Gold Gear to optimally price their vehicles based on the vehicle's parts using a prediction model. Before the creation of this project and the price prediction model, Gold Gear had no way of determining vehicle prices for outside markets beyond the United States. With this hole of information missing from Gold Gear, they weren’t ready to tackle their goals of expanding to new markets outside of the United States. However, with the use of this integrated program, Gold Gear can now reliably predict what a vehicle's price will be based on the vehicle parts that are used worldwide, thus giving them the power to predict any vehicle's price across the world. This is sure to help Gold Gear in its efforts to expand to new markets and increase Gold Gear's influence, competition, chance of market growth, and profits.

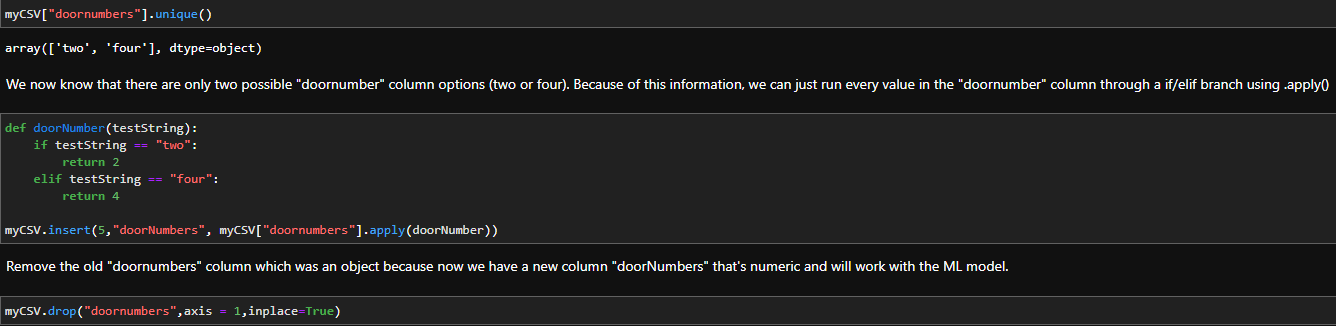
## **Datasets -**

All data used in the program was gathered/downloaded freely from Kaggle.com. The Python third-party library “pandas” is used to import this data and set it to the variable “myCSV”, as shown below. 

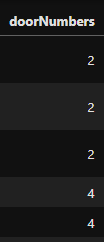
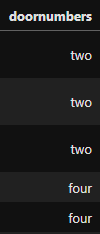
In addition, a sample of the raw, uncleaned data is shown below. (205 ROWS x 26 COLUMNS)

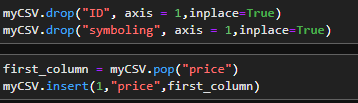
At first glance this raw, uncleaned data is a lot to take in and not very useful to Gold Gear. The data must be cleaned to be better visualized and usable with our machine-learning model. Note that because we are using a supervised learning model (linear regression), we need NUMERIC data, not CATEGORICAL data. Columns in the raw, uncleaned data like “doornumbers” will need to be transformed into numerical columns to be used by the algorithm. Before that, however, let’s first start cleaning the data by removing any potential duplicate entries/rows in the data. 

In the case of this specific dataset, there are no duplicate entries/rows since the number of rows/columns doesn’t change after the execution of this line of code. However, it’s always good practice to check and remove duplicate entries when working with data, so we will run it anyways. Next, let’s translate some categorical columns into numeric columns for the reason stated above.



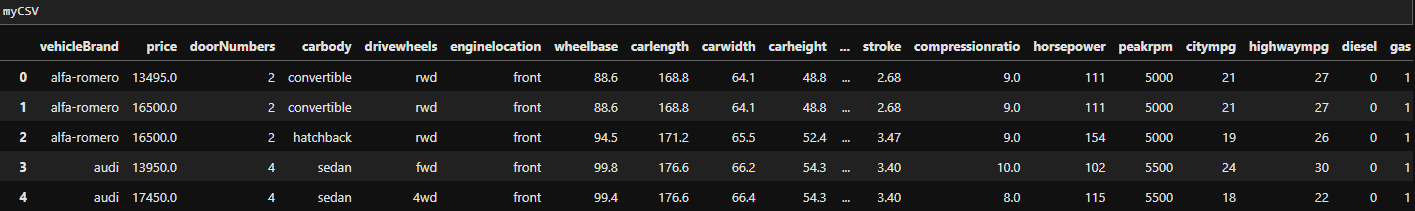
As we can see, first we need to identify the unique values associated with this column of data. In the case of the “doornumbers” column, a vehicle can have either two or four doors, which makes sense. Now we will simply create a function that takes a string input and returns the numerical correspondence and use “.apply()” to run the entire column of data through this function. All that is left is to remove the categorical “doornumbers” column, as we now have a numerical column that is more useful to us in terms of visualization and the machine-learning model.

BEFORE VS AFTER

This process is repeated on the “cylindernumber”, “fueltypes”, and “aspiration” columns. We then will remove the”ID” and “symboling” columns as they have no use to us. The “price” column, which is our target column, will be moved to the front of the DataFrame to promote better visibility of the value we’re primarily looking at. 

Lastly, we clean up the “name” column of the DataFrame which represents the vehicle's name. Before cleaning, the column had 147 unique values that represented the entire vehicle's name, like “Toyota Corolla 1200”. After cleaning, the column has 28 unique values that represent ONLY the brand name of the vehicle. Using the example above, the cleaned name would just be “toyota”.

This concludes the data-cleaning process, the following is a sample of the cleaned data.



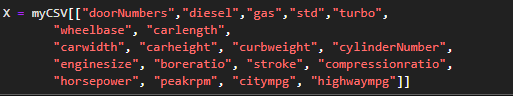
## **Data Product Code -**

The predictive method was constructed by setting up my “X” and “y” variables. The target variable “y” is what the model is attempting to predict, in this case, it is “price”.

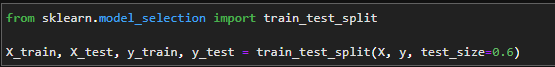


The “X” variable includes the features used to determine/predict the target variable (y variable).

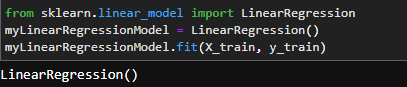
The X variable consists of all numeric columns. Categorical columns are not supported with regression algorithms, and more specifically, linear regression.



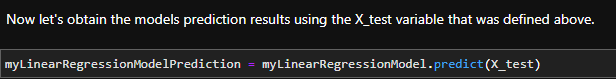
Once the target (y) and features (X) variables are assigned, the data needs to be split into two types of data, testing, and training. Data from the train set will be given to the prediction model which will predict the price that will be compared to the test set to see how close/far the prediction was.



As we can see, the “test\_size” is 0.6 (60%), meaning we’re using 60% of the data as a sample size. This number would normally be lower (around 30%), but to ensure a semi-accurate prediction, it has been raised. Now that we have our training and testing variables sorted, we can train our model using “fit()” and include our training variables as arguments.



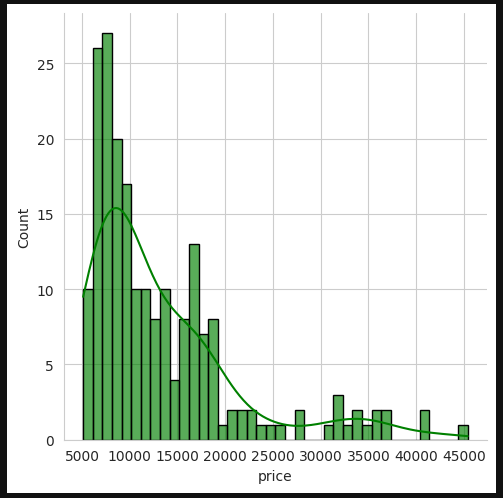
Finally, we set the prediction results to a variable for future comparisons with the actual correct values (y\_test)



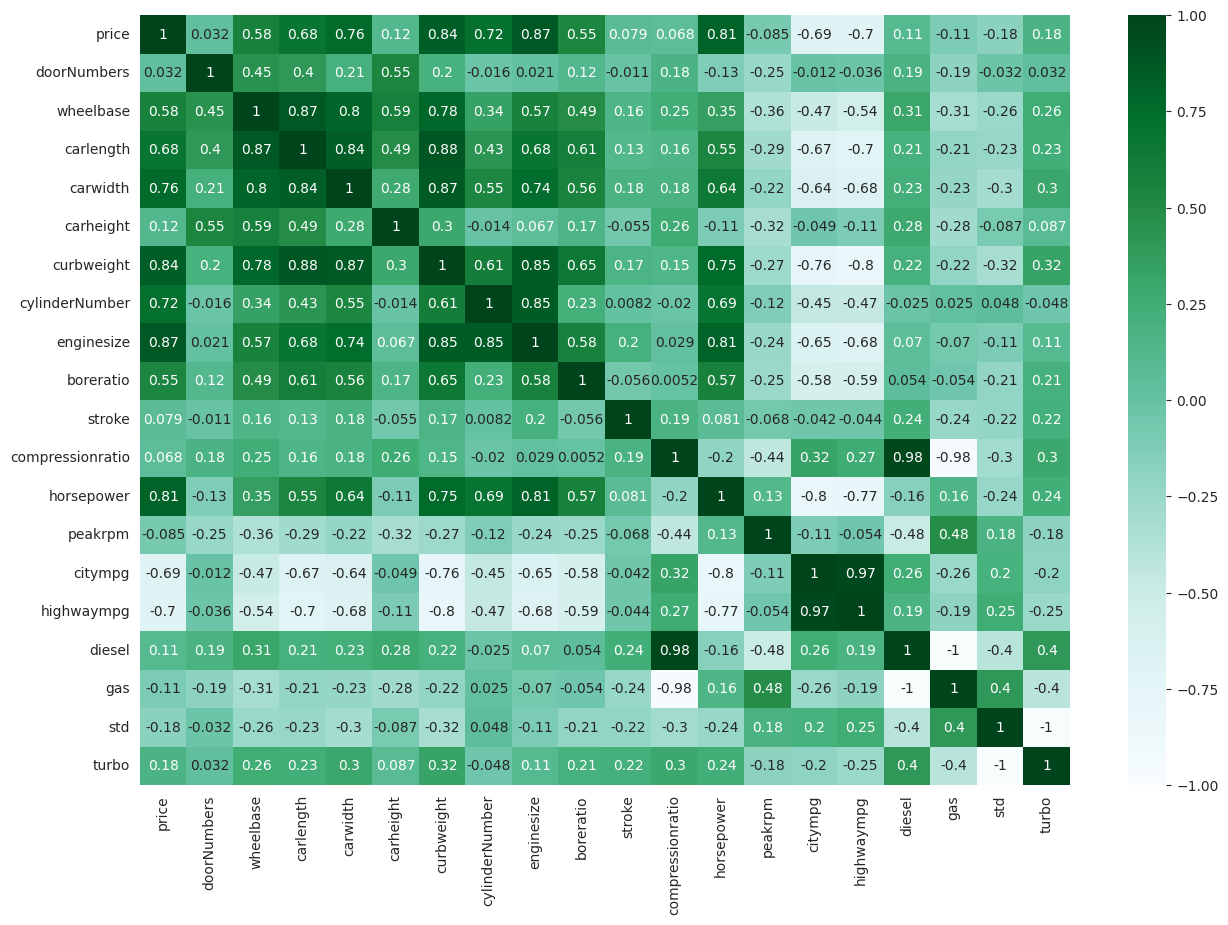
## **Hypothesis Verification -**

As a reminder, our original hypothesis was that by creating a price prediction model using vehicle parts through Linear Regression, reliable total vehicle price predictions could be made. Based on the results and the finished product, the hypothesis was mostly correct. When using the prediction model, a Root Mean Squared Error (RMSE) of around 4,000 is common, which meets the Evaluation plan requirements of 5,000 or below. This means the prediction model will be useful to Gold Gear, but a price prediction using the model could still be +4,000 or -4,000 off of the actual value. There is certainly room for improvement.

## **Effective Visualizations and Reporting -**

There are many visualizations displayed about the data and the relationships within the data in the finished product. All of the visualizations align with Gold Gears' interests, such as displaying the **distribution** of vehicles based on their price, as shown below. 

* This visualization represents the distribution of vehicles based on their price.
* As we can see, based on this data visualization, most vehicles cost between $5,000 - $20,000.

Other visual depictions of data include a **heatmap** that shows the relationship between all data on the CSV file. Useful for all types of information gathering, but when specifically looking at the price column, we gain an insight into what aspects of a vehicle (parts) play an important role in the overall price of a vehicle. 

For example, looking at the price column in the heatmap it looks like the following have a decent/high correlation (0.6+) with price.

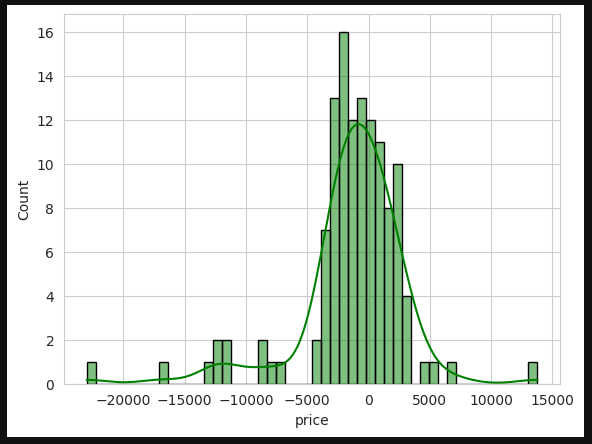
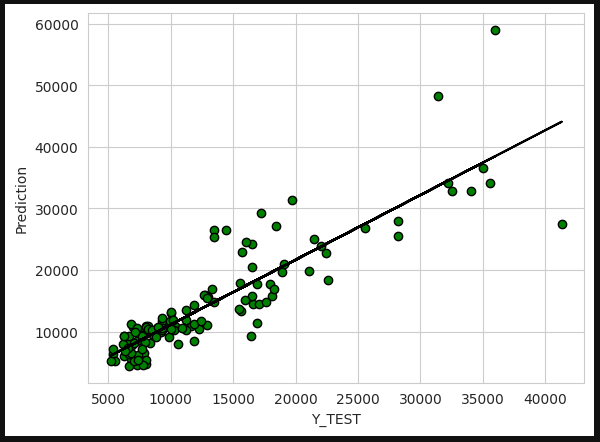
-HORSEPOWER (0.81) -ENGINESIZE (0.87) -CYLINDERNUMBER (0.72)

-CURBWEIGHT (0.84) -CARLENGTH (0.66) -CARWIDTH (0.7)

Besides those columns, the following have a medium correlation with price.

-BORERATIO (0.55) -WHEELBASE (0.58)

Visual representations of the data will also be shown after the model has undergone its predictions to better and clearly depict how far/close the model's predictions were. For example, the following **histogram** and **scatter plot** each display how off every prediction the model made was from the actual correct price.

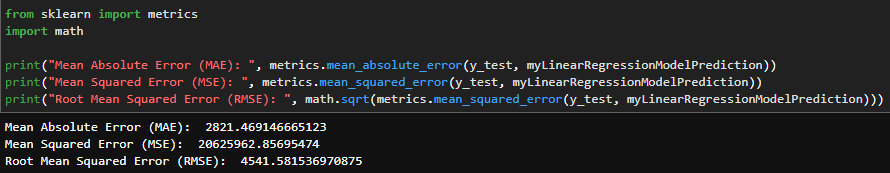


The entire program supports data preparation by cleaning and filtering the data when it’s first pulled to separate the important data that the company/prediction model needs from the more useless data that doesn’t help the company/prediction model do anything. This is shown above in the Datasets section. All of the data preparation/cleaning and the visualization of data are prevalent in the distribution plot, heatmap, histogram, and scatter plot.

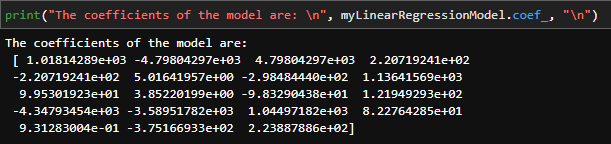
## **Accuracy Analysis -**

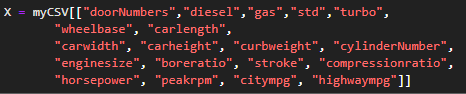
Scikit-Learn has many metric functions that are used to measure how accurate predictions are/end up being. Such metrics include the…

* Mean Absolute Error (MAE): This value represents the mean average of absolute values of errors. Won't punish large errors (outliers).
* Mean Squared Error (MSE): The superior version of MAE because it takes into account large errors (outliers).
* Root Mean Squared Error (RMSE): Most popular, same as above but the value is squared. This is the value that we use to gauge the performance and accuracy of the prediction model. Represents on average how far away a prediction is from the correct value.

All of these metrics are included as part of the post-prediction information-gathering process. 

Taking a look at the Root Mean Squared Error (RMSE), it gives us a value of around 3500-4500, which when looking at the histogram above makes sense because these two values mean the same thing. The RMSE value represents how off a prediction was ON AVERAGE. In this case, it means that a prediction of a vehicle's price (on average) could be off by around $3,500 - $4,500.

The coefficients are also displayed, which describes the relationship between the prediction features and the response value/prediction.



Taking a look at these coefficients, the higher the number the larger role the corresponding index on “X” plays in influencing the prediction model. For example, the fourth to last coefficient value (index = 16) would be linked to the “horsepower” column of data in the features list (index = 16) and because this value is particularly high, it greatly influences the value of the final prediction. The heatmap visualization shown above shows some of the same relationships.

## **Application Testing -**

Because of the nature of the application used, Jupyter Notebooks, Unit Testing is used on every cell of code. A cell of code is typically 1-3 lines of Python code, which is run, and any results/output of code is printed before the next cell is run. Unit testing like this in Jupyter Notebooks allows us to verify the behavior of smaller segments of code, ensuring functionality on a smaller and eventually larger scale.

During the implementation/development step of the project's lifecycle Regression Testing was also performed. Whenever a new cell of code was made and tested, the entire Jupyter Notebook was cleared of output and re-ran to confirm that the changes haven’t introduced new issues or bugs. This testing also goes hand in hand with Integration Testing, which was also done.

## **Application Files -**

There are only four application files used in this project. One of them is the Jupyter Notebook file (“WGU-C964-ComputerScienceCapstoneV2.ipynb”) which contains all the source code for the project. The other is a simple CSV file (“scrap price.csv”) that holds the raw data used during the project. This is the data that will be read into the Jupyter Notebooks environment using pandas. The “requirements.txt” file displays all the versions of third-party libraries used in the project and is used primarily for the Binder environment. The “README.md” file contains the Binder Badge button used to automatically run the development environment, use this if the link below doesn’t work.



The Jupyter Notebook will also be hosted using Binder to mimic a web application environment. Access to this web application will be as simple as clicking a link. (<https://mybinder.org/v2/gh/Xenonous/C964-Capstone/main?labpath=WGU-C964-ComputerScienceCapstoneFinal.ipynb>)

By using the Binder link, the entire Jupyter Notebook and all dependencies/libraries will be automatically downloaded. These include all third-party libraries used in the project like numpy, pandas, matplotlib, seaborn, sci-kit, and ipywidgets. A zipped file of the Jupyter Notebook and the CSV file will be attached as well. The file hierarchy for this zipped file is as described below.

./C964-Capstone-main

C964-Capstone-main\WGU-C964-ComputerScienceCapstoneFinal.ipynb

C964-Capstone-main\scrap price.csv

requirements.txt

README.md

## **User’s Guide -**

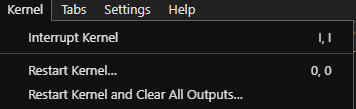
The following are the steps needed to access and run the application.

* **Step 1**: Access the Jupyter Notebook through the Binder environment by clicking this link and waiting for the environment to load. This will automatically load all dependencies and third-party libraries.

(<https://mybinder.org/v2/gh/Xenonous/C964-Capstone/main?labpath=WGU-C964-ComputerScienceCapstoneFinal.ipynb>)

* **Step 2**: Once the Jupyter Notebook is loaded, clear the Kernel of all potential outputs.

To do this go to Kernel -> Restart Kernel and Clear All Outputs…



* **Step 3**: Once the Jupyter Notebook is loaded and the Kernel/Output cells have been reset, SELECT THE FIRST CELL (imports) and RUN IT (SHIFT+ENTER).

You may also run a cell by clicking the run button. (looks like an arrow) 

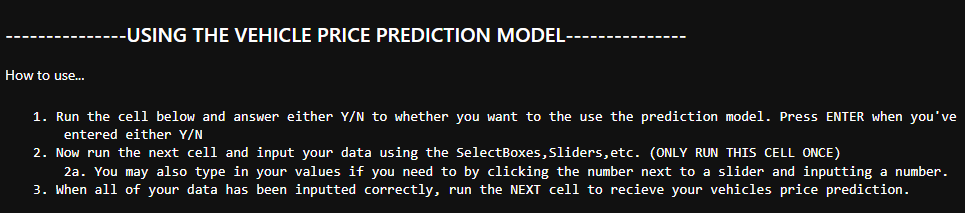
NOTE

[\*] - This means the cell is currently running, wait for it to finish before running the next cell to avoid bottlenecking the kernel. 

-CURRENTLY RUNNING



-DONE

* **Step 4**: After running the first cell, you’ll automatically select the next cell, so at this point, just keep running each cell (SHIFT+ENTER) and observing the output. Be careful not to run a selected cell twice unless specifically told you can.
* **Step 5**: Once you get to the VEHICLE PRICE PREDICTION MODEL section shown below, follow the instructions to input your values to get a price prediction.

## **Summation of Learning Experience -**

Prior experience with Python helped tremendously in tackling this project. WGU courses like Data Structures and Algorithms I (C949) and Data Structures and Algorithms II (950) helped in understanding and practice the basics of Python. The Business of IT courses helped in understanding key business methodologies and terminology that I can expect to see in the real world and were especially helpful during the process of writing this write-up. Before starting this project my knowledge of machine learning was limited and likely insufficient to do anything meaningful project-wise. To fill this knowledge gap, I completed an invaluable Udemy course on Data Science and Machine Learning that covered many Python third-party libraries used in the project such as numpy, pandas, matplotlib, and seaborn.

The experience of completing not just this project, but all projects at WGU has indubitably helped in developing and improving new skills, such as

* **Problem-solving skills**: Identifying and solving fairly complex problems. This includes things like meeting software requirements, debugging code, performance optimizations, etc. Problem-solving skills like these can be applied to many aspects of life like analytical reasoning and rational decision-making.
* **Technical skills**: No matter the project, new programming languages, tools, libraries, etc were learned. This process eventually increased my overall knowledge base, making learning more skills easier and contributing to a lifelong learning process.
* **Continuous, iterative development/improvement**: Through the process of creating many projects it's becoming apparent that software development is an iterative process that requires constant improvement. Learning from mistakes, reflecting, and taking in feedback are all a part of the software development process, just as the same process can be applied to learning as a whole.
* **Much more**: Project Management skills, organizational skills, collaboration and teamwork skills, etc.

# **SOURCES**

**NONE**