**Machine Learning Engineer Nanodegree**

Capstone Project

Himanshu Shekhar

September 27th, 2017

**Definition**

**Project Overview**

I will be working on Kaggle competition “New York City Taxi Trip Duration” where in I will build a model that predicts the total ride duration of taxi trips in New York City. This problem statement is relevant to transportation industry and can be applied in different contexts like ride sharing services, food/grocery delivery. Being able to predict ride duration accurately is extremely important for both the entities (service provider & customers) involved in the transaction. From service provider standpoint, it’s important as ride duration would dictate pricing that could be charged to a customer. In addition, predictability (duration) also provides good estimate of the number of drivers who would be present in certain region at any given time which is important so that customers could be matched with drivers/service provider in an efficient manner. From customer standpoint, predicting duration could help them make a decision as to when is the optimal time to start their commute.

With the advent of technology based cab services, this problem has become particularly important for the drivers and the customers. A good prediction mechanism can be instrumental for drivers in optimizing their returns, while also saving the customers from the uncertainties attached to a trip.

**Problem Statement**

Objective is to predict the time duration (in secs./mins.) of a New York taxi ride as a function of independent attributes like pick up and drop off location, time, volume etc. I will study the impact of various features and also attempt to find the best possible ways to leverage those features. I will explore Ensemble (decision tree/random forest) models to do the prediction.

**Metrics**

I will use Root Mean Square Error (RMSE) as a metric to measure model’s accuracy. To validate the usefulness of model, RMSE values for baseline model and the actual model developed would be compared on the test set. It has the advantage of being convex and physically interpretable (it has the same unit as time for duration prediction)

RMSE=1n∑i=1n(yi−y^i)2−−−−−−−−−−−−√

**Analysis**

**Data Exploration**

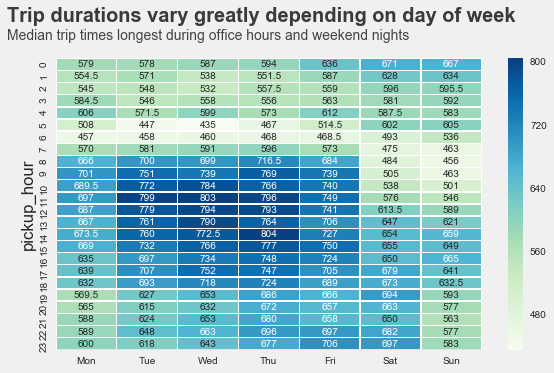
The data provides the details of taxi rides in the New York City from Jan 2016 to June 2016. This data is provided by the NYC Taxi and Limousine Commission (downloaded from Kaggle). Each trip records fields:

|  |  |  |
| --- | --- | --- |
| **#** | **Field** | **Description** |
| 1 | id | a unique identifier for each trip |
| 2 | vendor\_id | a code indicating the provider associated with the trip record |
| 3 | pickup\_datetime | date and time when the meter was engaged |
| 4 | dropoff\_datetime | date and time when the meter was disengaged |
| 5 | passenger\_count | the number of passengers in the vehicle (driver entered value) |
| 6 | pickup\_longitude | the longitude where the meter was engaged |
| 7 | pickup\_latitude | the latitude where the meter was engaged |
| 8 | dropoff\_longitude | the longitude where the meter was disengaged |
| 9 | dropoff\_latitude | the latitude where the meter was disengaged |
| 10 | store\_and\_fwd\_flag | This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server ( Y=store and forward; N=not a store and forward trip) |
| 11 | trip\_duration | duration of the trip in seconds |

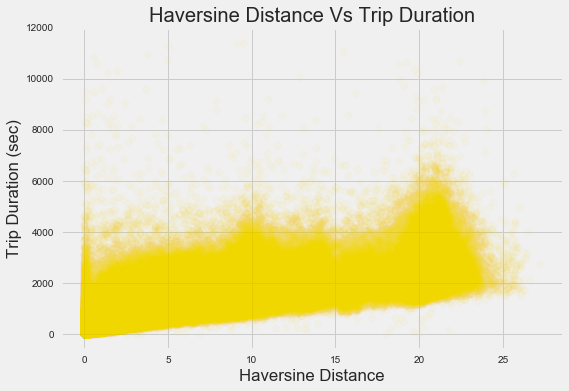
* From a preliminary analysis, it is evident that the target variable has some unusually high values so I will get rid of those to make the model less vulnerable to these outliers. Also, trip\_duration is highly skewed (right tailed) so I will apply a logarithmic transformation on this feature to reduce the range of values.
* The passenger\_count variable has a minimum value of 0 passengers, which is not relevant in the context of this usecase. These observations are most likely errors and will need to removed from the dataset.
* Based on different coordinate estimates of New York City, the latitude and longitude ranges are as follows: Latitude is between 40.7128 and 40.748817 whereas Longitude is between - 74.0059 and - 73.968285. The statisical summary of pick-up and drop-off coordinates show max and min observations that fall outside of the NYC city coordinate range. I will exclude these data points as this analysis is limited to New York City.

**Exploratory Visualization**

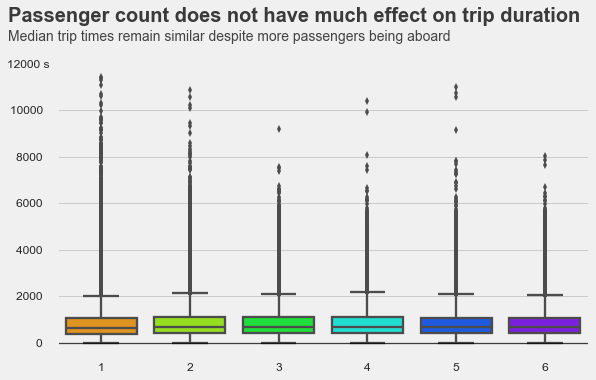
* **Time of day / Day of week:** The time of the day is an important factor to determine the duration a trip can take. The same trip during rush hours can take much longer compared to non-rush hours. The day of the week can have a significant impact on the predictions because we expect the weekdays to be more congested than the weekends especially during the day time. We can clearly observe from below heatmap the peak hours:
  + Monday - Friday : 8:00am - 6:00pm which is the ususal office hours
  + Thursday, Friday, Saturday Nights: 6:00 pm – midnight
  + Early Saturday & Sunday Mornings: 12:00 am - 1:00 am
  + Sunday Afternoons: 2:00 pm and 4:00 pm



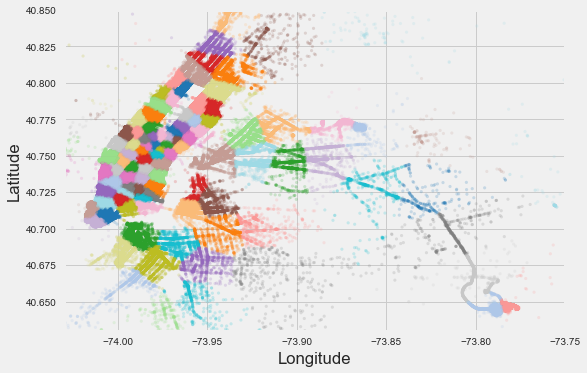
* **Distance**: I computed the haversine distance which determines the great-circle distance between two points on a sphere given their longitudes & latitudes. Below plot demonstrates a positive correlation between trip distance and duration



* **Passenger count:** The passenger\_count variable is the number of passengers in the vehicle as inputed by the driver. My assumption was that trips with more passengers are inherently longer due to more stops. However, as we can observe in below figure, median trip\_duration does not vary much as passenger\_count increases.

****

* **Location**: An important aspect of this prediction task is the pick-up and drop-off locations. As discussed above, the distance factor is indeed import to determine the duration but it is also important to consider the exact route the taxi is taking. Though we might not have the exact routes of the trip but the pick-up and drop-off coordinates can act as a proxy for this feature. An important aspect of this problem was to use the pickup and drop-off coordinates effectively. Directly feeding them to the model would have made less sense as the coordinates could have potentially infinite possible values and with the limited number of cases given in the training set, it would have been difficult to extract relevant patterns. To tackle this problem, I used kmeans clustering to create 62 clusters using the pick-up and drop-off coordinates.

****

**Algorithms and Techniques**

**K-Means**

I would use K-means algorithm to cluster the pick up & drop off longitude & latitude. The k-means algorithm divides a set of N samples X into K disjoint clusters C, each described by the mean \mu_j of the samples in the cluster. The means are commonly called the cluster “centroids”; note that they are not, in general, points from X, although they live in the same space. The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum of squared criterion:

\sum_{i=0}^{n}\min_{\mu_j \in C}(||x_j - \mu_i||^2)

Inertia, or the within-cluster sum of squares criterion, can be recognized as a measure of how internally coherent clusters are.

**Random Forest**

As traffic is clustered and aggregated more densely to different locations at different times, the location of the ride will clearly have an affect on the trip duration. Although there is no straightforward way of considering all locations between the start and end points of a ride, the pickup and dropoff locations are available in the dataset and can be used to model some of the effect of traffic and conjunctions. As traffic is clearly not varying solely based on the magnitude of the coordinates, the linear models fail to account for the nonlinear effect the locations have on traffic and hence trip duration. An algorithm that can better account for these nonlinearities is the random forest.

The random forest algorithm aggregates many decision trees built on bootstrapped samples of the training data in order to reduce the high variance of a single decision tree and improve prediction accuracy [1][2]. Each of these decision trees aims to divide the predictor space, i.e. the set of all

possible values for the features x1, x2, ..., xn , in J distinct and non-overlapping regions R1,R2, ...,RJ . The predictor space is divided into high-dimensional rectangles, with the goal to

find rectangles R1,R2, ...,RJ that minimize the RSS,

Σ Jj=1Σi∈Rjy ( (i) −y︿Rj)2(3) where is the mean response for the training observations y︿

Rj within the jth rectangle. When building each tree, a top-down approach is taken. Beginning with all points in the same region, the algorithm successively splits the predictor space into two halves, stopping when there are no more than five points in a region. At each split, a prediction xj and cutpoints are chosen such that splitting the predictor space into the regions {x | xj < s} and {x | xj ≥ s} leads to the biggest reduction in RSS. Defining the pair of halves as R1(j, s) and R2(j, s) , at each split we seek to find j and s that minimize the equation

Σ (4)i: x(i)∈R1(j, s)y ( (i) −y︿Rj)2+Σi: x(i)∈R2(j, s)y ( (i) −y︿Rj)

Once the regions are defined, a prediction by a single tree is made by averaging the responses of the training observations in the region to which the test observation belongs. In the random forest, a large number of trees are fit, each using a bootstrap sample from the training data, and a prediction of a

new observation is made using the mean of the predictions by all the trees. At each split, only m of the total n predictors are randomly chosen to be considered. This approach is taken to decorrelate the trees, as considering all predictors might yield very similar trees when one or a few predictors are

particularly strong. As averaging many uncorrelated trees leads to a larger reduction in variance, this approach often yields better prediction results.

**Benchmark**

As a baseline prediction, I ran linear regression to do the prediction. The linear regression model finds the set of coefficients θ that minimize the sum of squared errors

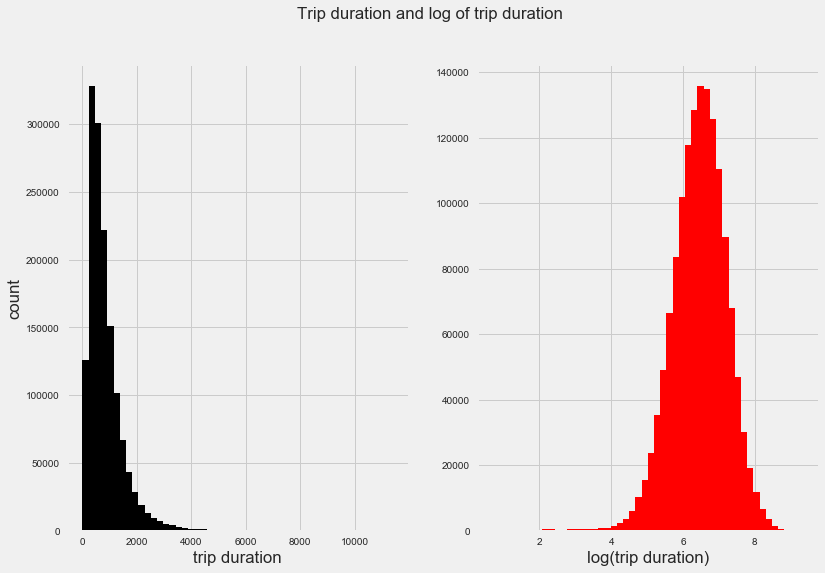
y(i) = θ0 + Σ xjθj j(i)

The RMSE for the Baseline model on the test set was obtained as 0.48 for trip\_duration and R2\_score of 0.61.

**Methodology**

**Data Preprocessing**

* **Trip Duration -** We can clearly observe that trip\_duration is highly skewed to the right. So I applied log transformation to trip\_duration which will normalize its distribution (right graph) and reduce the influence of the high observations in the right tail.



* **Outlier treatment/abnormal observations –** 
  + Removed observations which had 0 passengers.
  + Capped the observations within 2 standard deviation of trip\_duration.
  + Restricted observations based on coordinate estimates of New York City
    - Latitude is between 40.7128 and 40.748817
    - Longitude is between - 74.0059 and - 73.968285

**Implementation**

* Representing the pick-up and drop-off coordinate data is an important aspect for prediction as it captures the location dynamics. I opted for creating clusters on the given pick-up and dropoff coordinates. To represent the pickup and drop location, the latitude, longitude pairs were partitioned into clusters. The choice for the number of clusters used for this task was done in a simplistic manner i.e. NY region can be sub-divided into 62 counties so I used that as a frame of reference when choosing the number of clusters
* **Extracted features** like pickup\_weekday, pickup\_hour, pickup\_month from pickup\_datetime. As observed in exploratory visualization section, trip\_duration can vary depending upon the day/time so including these independent features into the model would help us explain the target variable.
* As an initial pass (unoptimized model), I created the regressor object with RandomForestRegressor with default values of the parameters and fit it on the training data

**Refinement**

* In order to improve the accuracy of the unoptimized model, I will tweak below listed parameters:
  + **n\_estimators:** This is the number of trees you want to build before taking the maximum voting or averages of predictions. Higher number of trees give you better performance but makes your code slower (default = 10)
  + **max\_features:** These are the maximum number of features Random Forest is allowed to try in individual tree (default = ‘auto’ i.e. use all features)
  + **min\_samples\_leaf:** The minimum number of samples required to be at a leaf node (default =1)
  + **min\_samples\_split:** The minimum number of samples required to split an internal node (default = 2)
* Below table captures various iterations I ran on the model for different parameter values:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **n\_estimators** | **max\_features** | **min\_samples\_leaf** | **min\_samples\_split** | **RMSE** |
| 1 | 10 | auto | 1 | 2 | 0.3863 |
| 2 | 10 | auto | 50 | 75 | 0.3788 |
| 3 | 10 | sqrt | 50 | 75 | 0.3864 |
| 4 | 50 | auto | 50 | 75 | 0.3773 |
| 5 | 100 | auto | 50 | 75 | 0.3771 |
| **6** | **150** | **auto** | **50** | **75** | **0.377** |
| 7 | 50 | sqrt | 50 | 75 | 0.3838 |
| 8 | 100 | sqrt | 50 | 75 | 0.3838 |
| 9 | 150 | sqrt | 50 | 75 | 0.3836 |

We know that the validation RMSE will tend to reduce as we increase the parameter value of n estimators. However, we need to choose a reasonable value of n estimators beyond which the

validation RMSE does not significantly reduce. To see this, observe in below table that RMSE value decreases from 0.3863 to 0.377 as we increased the value of estimators from 10 to 150. However, there is only a marginal improvement in RMSE when we change the estimator value from 100 to 150 so I chose to stop at this point.

**Results**

**Model Evaluation & Validation**

During development, a validation set was used to evaluate the model. The final hyperparameters were chosen because they performed the best among the tried combinations. Here is the final list of parameters for the chosen model

* N\_estimators: 150
* Max\_features: ‘auto’
* Min\_samples\_leaf: 50
* Min\_samples\_split: 75

In order to test the robustness of the model, I applied K fold cross validation strategy on the model.

In *k*-fold cross-validation, the original sample is randomly partitioned into *k* equal sized subsamples. Of the *k* subsamples, a single subsample is retained as the validation data for testing the model, and the remaining *k* − 1 subsamples are used as training data. The cross-validation process is then repeated *k* times (the *folds*), with each of the *k* subsamples used exactly once as the validation data. The *k* results from the folds can then be averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling (see below) is that all observations are used for both training and validation, and each observation is used for validation exactly once. With K (3) in this case, I obtained mean RMSE of 0.379 which is consistent with the score(0.377) we obtained earlier.

Most important features observed for Duration prediction are distance, pickup\_hour, center latitude, center longitude and pickup\_weekday. From Random forest, we obtained the importance of features as follows:

|  |  |
| --- | --- |
| **Feature** | **Random Forest Feature Importance** |
| Haversine\_distance | 0.8228 |
| Pickup\_hour | .0575 |
| Center latitude | .0416 |
| Center longitude | .0291 |
| Pickup weekday | .0215 |
| Dropoff\_cluster | .0087 |
| Vendor\_id | .0073 |

**Justification**

The random forest model outperform all other models used, as it manages to model the nonlinearities of traffic and location effect. Although the model accounts for the effect of pickup and dropoff locations, it has no way of modeling the effects of the locations along the route. A ride between two locations with high traffic can still be relatively fast if it goes through high-speed areas with little or no traffic. To account for the route, I computed the center latitude and longitude and fed that as independent factor in the model. Considering what is accounted for in the models, they are believed to predict duration relatively precisely.

|  |  |  |  |
| --- | --- | --- | --- |
| **Performance Metric** | **Benchmark Model** | **Unoptimized Model** | **Optimized Model** |
| RMSE | 0.56 | 0.3863 | 0.3770 |

Performance metric of optimized model is significantly better than the benchmark model.

**Conclusion**

**Reflection**

The process used for this project can be summarized using the following steps:

1. An initial problem and relevant, public datasets were found (Kaggle)

2. The data was downloaded and preprocessed for feature engineering

3. A benchmark (linear regression model) was created for prediction

4. The optimized model was trained with RandomForestRegressor. It was trained using the data (multiple times, until a good set of parameters were found)

5. Implemented k-fold cross validation strategy to test the robustness of the model

One of the most interesting aspect of the project was feature engineering . Challenge was to make effective use of the location co-ordinates without diluting its value. As explained in earlier section, I used k-means clustering to effectively capture the effect of pick up & drop off coordinate. Also to capture on route dynamics, I computed center longitude & latitude and included them as independent factor in the model (assumption here is that taxi would pass through this point to reach their final destination)

**Improvement**

Below are some ideas to improve upon the performance of the model which can essentially be categorized into:

* **Feature Generation**
  + Making use of **weather information** could help us explain the trip duration as snowfall, rainfall among others dictate the speed at which a vehicle is moving.
  + Similarly, having access to fastest route, second fastest route and path could help us explain the trip duration. This dataset can be generated from **Open Source Routing Machine.**
  + **Coordinate Transformation :** To further model the effect of the pickup and dropoff locations we could transform the coordinates [9]. Most of the streets and avenues in Manhattan are aligned in a grid structure. With the hypothesis that the avenue or street could explain some of the effect of the location, transforming the coordinates so that the splits in the random forest algorithm will be made aligned and perpendicularly to the avenues and streets, could potentially yield better predictions.
* **Algorithm**
  + We could develop a model based on **neural nets / Multilayer perceptrons** and compare it’s performance against Random Forest to assess whether it gives better output

**References**

[1] James, G, D Witten, T Hastie, and R Tibshirani. *An introduction to statistical learning*. Vol. 6. , New York, Springer., 2013.

[2] Friedman, J, T Hastie, and R Tibshirani. *The elements of statistical learning*. Vol. 1. , Berlin, Springer, 2001.

[3] Blaser, Rico, and Piotr Fryzlewicz. "Random rotation ensembles." *J Mach Learning Res* 2 (2015): 1-15.

**[4]<http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>**

**Data source:** [**https://www.kaggle.com/c/nyc-taxi-trip-duration/data**](https://www.kaggle.com/c/nyc-taxi-trip-duration/data)

**I. Definition**

*(approx. 1-2 pages)*

**Project Overview**

In this section, look to provide a high-level overview of the project in layman’s terms. Questions to ask yourself when writing this section:

* *Has an overview of the project been provided, such as the problem domain, project origin, and related datasets or input data?*
* *Has enough background information been given so that an uninformed reader would understand the problem domain and following problem statement?*

**Problem Statement**

In this section, you will want to clearly define the problem that you are trying to solve, including the strategy (outline of tasks) you will use to achieve the desired solution. You should also thoroughly discuss what the intended solution will be for this problem. Questions to ask yourself when writing this section:

* *Is the problem statement clearly defined? Will the reader understand what you are expecting to solve?*
* *Have you thoroughly discussed how you will attempt to solve the problem?*
* *Is an anticipated solution clearly defined? Will the reader understand what results you are looking for?*

**Metrics**

In this section, you will need to clearly define the metrics or calculations you will use to measure performance of a model or result in your project. These calculations and metrics should be justified based on the characteristics of the problem and problem domain. Questions to ask yourself when writing this section:

* *Are the metrics you’ve chosen to measure the performance of your models clearly discussed and defined?*
* *Have you provided reasonable justification for the metrics chosen based on the problem and solution?*

**II. Analysis**

*(approx. 2-4 pages)*

**Data Exploration**

In this section, you will be expected to analyze the data you are using for the problem. This data can either be in the form of a dataset (or datasets), input data (or input files), or even an environment. The type of data should be thoroughly described and, if possible, have basic statistics and information presented (such as discussion of input features or defining characteristics about the input or environment). Any abnormalities or interesting qualities about the data that may need to be addressed have been identified (such as features that need to be transformed or the possibility of outliers). Questions to ask yourself when writing this section:

* *If a dataset is present for this problem, have you thoroughly discussed certain features about the dataset? Has a data sample been provided to the reader?*
* *If a dataset is present for this problem, are statistics about the dataset calculated and reported? Have any relevant results from this calculation been discussed?*
* *If a dataset is****not****present for this problem, has discussion been made about the input space or input data for your problem?*
* *Are there any abnormalities or characteristics about the input space or dataset that need to be addressed? (categorical variables, missing values, outliers, etc.)*

**Exploratory Visualization**

In this section, you will need to provide some form of visualization that summarizes or extracts a relevant characteristic or feature about the data. The visualization should adequately support the data being used. Discuss why this visualization was chosen and how it is relevant. Questions to ask yourself when writing this section:

* *Have you visualized a relevant characteristic or feature about the dataset or input data?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

**Algorithms and Techniques**

In this section, you will need to discuss the algorithms and techniques you intend to use for solving the problem. You should justify the use of each one based on the characteristics of the problem and the problem domain. Questions to ask yourself when writing this section:

* *Are the algorithms you will use, including any default variables/parameters in the project clearly defined?*
* *Are the techniques to be used thoroughly discussed and justified?*
* *Is it made clear how the input data or datasets will be handled by the algorithms and techniques chosen?*

**Benchmark**

In this section, you will need to provide a clearly defined benchmark result or threshold for comparing across performances obtained by your solution. The reasoning behind the benchmark (in the case where it is not an established result) should be discussed. Questions to ask yourself when writing this section:

* *Has some result or value been provided that acts as a benchmark for measuring performance?*
* *Is it clear how this result or value was obtained (whether by data or by hypothesis)?*

**III. Methodology**

*(approx. 3-5 pages)*

**Data Preprocessing**

In this section, all of your preprocessing steps will need to be clearly documented, if any were necessary. From the previous section, any of the abnormalities or characteristics that you identified about the dataset will be addressed and corrected here. Questions to ask yourself when writing this section:

* *If the algorithms chosen require preprocessing steps like feature selection or feature transformations, have they been properly documented?*
* *Based on the****Data Exploration****section, if there were abnormalities or characteristics that needed to be addressed, have they been properly corrected?*
* *If no preprocessing is needed, has it been made clear why?*

**Implementation**

In this section, the process for which metrics, algorithms, and techniques that you implemented for the given data will need to be clearly documented. It should be abundantly clear how the implementation was carried out, and discussion should be made regarding any complications that occurred during this process. Questions to ask yourself when writing this section:

* *Is it made clear how the algorithms and techniques were implemented with the given datasets or input data?*
* *Were there any complications with the original metrics or techniques that required changing prior to acquiring a solution?*
* *Was there any part of the coding process (e.g., writing complicated functions) that should be documented?*

**Refinement**

In this section, you will need to discuss the process of improvement you made upon the algorithms and techniques you used in your implementation. For example, adjusting parameters for certain models to acquire improved solutions would fall under the refinement category. Your initial and final solutions should be reported, as well as any significant intermediate results as necessary. Questions to ask yourself when writing this section:

* *Has an initial solution been found and clearly reported?*
* *Is the process of improvement clearly documented, such as what techniques were used?*
* *Are intermediate and final solutions clearly reported as the process is improved?*

**IV. Results**

*(approx. 2-3 pages)*

**Model Evaluation and Validation**

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model’s solution is affected (this is called sensitivity analysis). Questions to ask yourself when writing this section:

* *Is the final model reasonable and aligning with solution expectations? Are the final parameters of the model appropriate?*
* *Has the final model been tested with various inputs to evaluate whether the model generalizes well to unseen data?*
* *Is the model robust enough for the problem? Do small perturbations (changes) in training data or the input space greatly affect the results?*
* *Can results found from the model be trusted?*

**Justification**

In this section, your model’s final solution and its results should be compared to the benchmark you established earlier in the project using some type of statistical analysis. You should also justify whether these results and the solution are significant enough to have solved the problem posed in the project. Questions to ask yourself when writing this section:

* *Are the final results found stronger than the benchmark result reported earlier?*
* *Have you thoroughly analyzed and discussed the final solution?*
* *Is the final solution significant enough to have solved the problem?*

**V. Conclusion**

*(approx. 1-2 pages)*

**Free-Form Visualization**

In this section, you will need to provide some form of visualization that emphasizes an important quality about the project. It is much more free-form, but should reasonably support a significant result or characteristic about the problem that you want to discuss. Questions to ask yourself when writing this section:

* *Have you visualized a relevant or important quality about the problem, dataset, input data, or results?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

**Reflection**

In this section, you will summarize the entire end-to-end problem solution and discuss one or two particular aspects of the project you found interesting or difficult. You are expected to reflect on the project as a whole to show that you have a firm understanding of the entire process employed in your work. Questions to ask yourself when writing this section:

* *Have you thoroughly summarized the entire process you used for this project?*
* *Were there any interesting aspects of the project?*
* *Were there any difficult aspects of the project?*
* *Does the final model and solution fit your expectations for the problem, and should it be used in a general setting to solve these types of problems?*

**Improvement**

In this section, you will need to provide discussion as to how one aspect of the implementation you designed could be improved. As an example, consider ways your implementation can be made more general, and what would need to be modified. You do not need to make this improvement, but the potential solutions resulting from these changes are considered and compared/contrasted to your current solution. Questions to ask yourself when writing this section:

* *Are there further improvements that could be made on the algorithms or techniques you used in this project?*
* *Were there algorithms or techniques you researched that you did not know how to implement, but would consider using if you knew how?*
* *If you used your final solution as the new benchmark, do you think an even better solution exists?*