|  |
| --- |
| Gallogly College of Engineering, University of Oklahoma |
| Petabyte Pirates Project Report |
| Link to Checkpoint Review Presentation: <> |

|  |
| --- |
| ACS 5513 – Machine Learning Practice  Summer 2025 |

Contents

[Project Charter 2](#_Toc2061437022)

[Business background 3](#_Toc457265002)

[Scope 3](#_Toc1811187642)

[Personnel 3](#_Toc185125116)

[Metrics 3](#_Toc1103303309)

[Plan 4](#_Toc211694713)

[Architecture 4](#_Toc1784352023)

[Communication 4](#_Toc943437985)

[Data Report 4](#_Toc1997468383)

[Raw Data Sources 4](#_Toc1817396984)

[Data dictionary 4](#_Toc1634588477)

[Data report 4](#_Toc700013680)

[Software Architecture Document 4](#_Toc577306888)

[Model Report 4](#_Toc1610672590)

[Analytic Approach 4](#_Toc966041009)

[Model Description 4](#_Toc1690796134)

[Results (Model Performance) 4](#_Toc147836099)

[Model Understanding 4](#_Toc1936131506)

[Conclusion and Discussions for Next Steps 4](#_Toc479348042)

[Exit Report 4](#_Toc873147791)

[Overview 4](#_Toc554453309)

[Business Domain 4](#_Toc163256039)

[Business Problem 4](#_Toc1226034779)

[Data Processing 4](#_Toc1745329925)

[Modeling, Validation 4](#_Toc1862409350)

[Solution Architecture 4](#_Toc1594715550)

[Benefits 4](#_Toc1399844061)

[Company Benefit 4](#_Toc2117571128)

[Customer Benefit 4](#_Toc1037639037)

[Learnings 4](#_Toc427237381)

[Project Execution 4](#_Toc247812519)

[Data science / Engineering 4](#_Toc465387921)

[Domain 4](#_Toc838593852)

[Product 4](#_Toc1028251719)

[What's unique about this project, specific challenges 4](#_Toc1819678161)

[Links 4](#_Toc1396440720)

[Next Steps 4](#_Toc37280453)

[Appendix 4](#_Toc311174573)

# [Project Charter](https://github.com/Azure/Azure-TDSP-ProjectTemplate/blob/master/Docs/Project/Charter.md)

## Business Background

The process of home valuation is complex and multi-faceted, so much so that there are entire research firms dedicated to providing market intelligence to real estate companies, developers, and to end-users via Zillow and Redfin. **A Home Valuing Company Inc.** is a residential property valuation intelligence firm that provides data and analytics to appraisers and real-estate agents via proprietary APIs and custom-built SaaS web applications maintained by the client. The firm also provides an in-house, subscription-based web portal for homeowners to understand real estate comps based on home characteristics, as well as keep track of their home’s value over time.

## Business Problem

Underwriters use rule-of-thumb comparable and simplistic regression tools that miss nonlinear interactions between a house and its characteristics, such as neighborhoods, age, and structural components. This can create valuation errors that cost lenders & customers $1000s in losses.

## Scope

An end-to-end ML service that ingests the Ames, IA housing CSV (~2,900 rows) from a version-controlled source and logs ingestion metadata to a notebook. Raw data is cleaned by dropping high-missing columns and taming unusually large or small values, so they don’t skew the results. We then crate 4 new measures: house age, years since last remodel, total living area and a “quality” size score and put every number on the same scale so the model can learn effectively.

Next, we split the homes into a “teach” group and “check” group. The model learns patterns from the teach group, like how square footage and overall quality relate to the price and then we confirm its accuracy on the check group, tracking how close its guesses are to real sale price.

Finally, a simple web form lets users enter home details and instantly get a price estimate. Each query is recorded in a table where you can copy any row, tweak one feature and see side-by-side of a “what-if” comparisons. All requests are logged so we can monitor its performance.

## Personnel

* Project Lead – Sean Miller
* PM – Farhan Hassan
* Data scientist(s) - Dewayne Hafenstein, Sean Miller
* Account Manager – Farhan Hassan
* Data Administrator – Sean Miller
* Business contact – Dewayne Hafenstein

## Metrics

* The measure of success metric is to be able to predict the selling price within +/- 2% of its actual selling price 95% of the time.
* Improve the prediction of sale prices to maximize revenue for sellers and listing agents.
* The current national list-to-sale ratio is about 97% but varies by locality. In tight markets, this ratio exceeds 100% with sale prices exceeding the list price. This tool will account for these ratio differences and provide an estimate of what the actual selling price would be. The current list-to-sell pricing in Ames Iowa (2025) shows that 71.4% of the properties sell below their listing price (Zillow). The discounted rate appears to be about –1.9% on average in Ames Iowa.
* The data set will be split into training and test data. The training data will be used to create and train the initial model, and the test data will be used to verify the model's accuracy.

## Plan

**Phase 1**: Establish the problem, set our Goal(s) to accomplish, explore the data set, perform correlation analysis among the features to determine which features to focus on, and determine the target architecture.

**Phase 2**: Determine the exact model to use. At the time of Phase 1 we suspect it will be Logistic Regression or multiple linear regression. Data analysis will determine the actual algorithm used.

**Phase 3:** Operationalize a limited API and connect to a proof-of-concept web application.

## Architecture

We build on the Ames; IA housing dataset downloaded either directly via Kaggle or pulled from the Git repo and load it straight into memory with no external data movement tools. The entire solution is written in Python and deployed as two isolated Flask applications: one hosts the machine learning engine and model artifacts, and the other serves the user-facing web interface. By splitting these responsibilities, we give the model its own dedicated runtime (maximizing memory and compute for retraining or batch analysis) while shielding its complexity behind a clean RESTful API.

On the front end, an HTML/CSS/JavaScript form captures the 11 property features and submits them as JSON to the /predict end point. The backend service validates, and preprocesses inputs, runs the trained model and returns a price estimate. Each requests original inputs and predicted value are then logged to a results table in the browser.

## Communication

* The team stays in communication via Microsoft Teams, as well as virtual zoom meetings to ensure deliverables are equally distributed and progress is made across all fronts.

# Data Report

**Overview:** While the team was initially skeptical of the dataset, given its age, limited scope, and size, we were pleasantly surprised by the significant number of features and the capability to engineer new and impactful features for data modeling. Our tool of choice was Google’s Colab jupyter environment, which provides collaborative capabilities and dedicated VMs with plug-and-play runtimes suitable for speedy analysis.

**Note:** the comprehensive data report has sections that detail data quality, target variable, individual variables, rankings, and relationships. This is available in the project’s GitHub repository ([link](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md)).

### Raw Data Sources

| **Dataset Name** | **Original Location** | **Destination Location** | **Link to Source** |
| --- | --- | --- | --- |
| Ames Housing | This dataset is publicly available via Kaggle | A copy of the dataset is stored in the ACS5513 GitHub repo | Source: [Link](https://www.kaggle.com/datasets/prevek18/ames-housing-dataset)  GitHub: [Link](https://github.com/dewayneh57/ACS5513/blob/main/AmesHousing.csv) |

The raw Ames, Iowa housing dataset is publicly available via Kaggle, and contains the sales transaction records of over 2900 real estate transactions between January 2006 and July 2010. It was chosen due to its uniquely robust inclusion of 81 features and the target variable SalePrice. This is of critical importance for a holistic statistical evaluation and ML modeling project.

This dataset can be either downloaded directly from Kaggle, or by using the KaggleHub component (library) in python, programmatically as part of the ML engine start up.

**Applicable Data Report Sections:**

* [General Summary of the Data](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#general-summary-of-the-data)

**Applicable Notebook Sections:**

* [Initial Imports and Data Sourcing](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=5ztGCwnPQLrE&line=1&uniqifier=1)
* [Normal Distribution Testing](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=OrHMeXNmXYXV&line=3&uniqifier=1)

### Processed Data

| **Processed Dataset Name** | **Input Dataset(s)** | **Data Processing Tools/Scripts** | **Link to Report** |
| --- | --- | --- | --- |
| Processed Dataset 1 | Ames Housing | [acs\_5513\_petabyte\_pirates\_project\_deliverable\_1.py](https://github.com/dewayneh57/ACS5513/blob/main/acs_5513_petabyte_pirates_project_deliverable_1.py)  Jupyter Notebook: **Data Processing and EDA** section ([GitHub](https://github.com/dewayneh57/ACS5513/blob/main/ACS_5513_Petabyte_Pirates_Project_Deliverable_1.ipynb), [Colab](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=C1gpOGy1QTGD&line=1&uniqifier=1)) | [Deliverable 1 Report](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md) |

The data processing stage consisted of data wrangling and exploratory data analysis. Features were chosen to continue to the final data frame based on the following data quality heuristics:

* **Missing Data**: cannot account for over 60% of values in each feature.
* **Value Frequency**: one feature value cannot account for greater than 90% of transactions.
* **Engineered Features:** all features that have been engineered, such as transformed with ordinal maps or one-hot encoding, will be removed to eliminate redundancy.

**Applicable Data Report Sections:**

* [Data Quality Summary](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#data-quality-summary)
* [Target Variable - SalesPrice](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#target-variable--saleprice)

**Applicable Notebook Sections:**

* [Data Processing and EDA](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=C1gpOGy1QTGD&line=1&uniqifier=1)
* [Time Series Data Analysis](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=0Q81EyiZbO8E&line=3&uniqifier=1)
* [Categorical Exploratory Data Analysis](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=zQ34ltUVdnrR&line=1&uniqifier=1)

### Feature Sets

| **Feature Set Name** | **Input Dataset(s)** | **Feature Engineering Tools/Scripts** | **Link to Report** |
| --- | --- | --- | --- |
| df\_cleaned | Ames Housing (transformed during processing) | [acs\_5513\_petabyte\_pirates\_project\_deliverable\_1.py](https://github.com/dewayneh57/ACS5513/blob/main/acs_5513_petabyte_pirates_project_deliverable_1.py)  Jupyter Notebook: **Data Enrichment and Engineering** section ([GitHub](https://github.com/dewayneh57/ACS5513/blob/main/ACS_5513_Petabyte_Pirates_Project_Deliverable_1.ipynb), [Colab](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=prSVCMuU_hvq&line=5&uniqifier=1)) | [Deliverable 1 Report](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md) |

Numerical features were engineered based on their inherent characteristics, such as age, size and quality, binary (yes/no), and relative time when sold. Explanatory (or categorical) features were ordinally mapped based on their values, e.g. Ex (Excellent) = 5 and Po (Poor) = 1 (see [map](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=INN4YKxP5W-2&line=13&uniqifier=1)). Finally, nominal columns were one-hot encoded and evaluated for statistical significance. The final list of features was constructed using the following statistical heuristics:

* **Statistical Evaluation (Pearson)**: features that do not meet a correlation coefficient |r| >= 0.5 will be dropped from the final data frame.
* **One-Hot Encoding:** dummy features will be removed if they do not meet the bar for statistical significance as determined by the point-biserial correlation test.

**Applicable Data Report Sections:**

* [Individual Variables (Top Examples)](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#individual-variables-top-examples)
* [Variable ranking (numeric Pearson r with SalePrice)](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#variable-ranking-numeric-pearson-r-with-saleprice)
* [Individual Variables (Bottom Examples)](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#individual-variables-bottom-examples)
* [Relationship between Explanatory Variables and Target Variable](https://github.com/dewayneh57/ACS5513/blob/main/ames_data_report.md#relationship-between-explanatory-variables-and-target-variable)

**Applicable Notebook Sections:**

* [Data Enrichment and Engineering](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=prSVCMuU_hvq&line=5&uniqifier=1)
* [Final Output](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=qb7bRoSv_uuI&line=8&uniqifier=1)
* [Conducting Early Trials](https://colab.research.google.com/drive/10mTYa4TVJd4LEYcQDxOxoOGWro5ZIR_D#scrollTo=xb0KY_ZulX54&line=3&uniqifier=1)

# Data Dictionary

See full, living data dictionary [here](https://docs.google.com/spreadsheets/d/1zRmdRlc2efk0RiQ3xv9OARQlnGgF3sbfwbhnpNbcv7Y/edit?usp=sharing). Double click the cell in the ***feature\_and\_value\_description*** column (D) to expand the full description.

**Overall Quality**

|  |  |
| --- | --- |
| Description | Rates the overall material and finish of the house  10 Very Excellent  9 Excellent  8 Very Good  7 Good  6 Above Average  5 Average  4 Below Average  3 Fair  2 Poor  1 Very Poor |
| Data Type | Integer |
| Units | Subjective Rating, 1-10 |

**Year Built**

|  |  |
| --- | --- |
| Description | The year the house was initially constructed. |
| Data Type | Integer |
| Units | Years |

**Year remodeled or additions added**

|  |  |
| --- | --- |
| Description | The date of the most recent remodeling or addition was added to the house. If no remodels or additions were added, this would be the same as the year it was built. |
| Data Type | Integer |
| Units | Years |

**Total Basement Square Feet**

|  |  |
| --- | --- |
| Description | The total square foot area of any basement for the house. |
| Data Type | Integer |
| Units | Square feet |

**First Floor Square Feet**

|  |  |
| --- | --- |
| Description | The total living area of the first-floor rooms of the house. |
| Data Type | Integer |
| Units | Square feet |

**Above Grade Living Area**

|  |  |
| --- | --- |
| Description | The total living area in square feet for all areas above grade (ground) level. |
| Data Type | Integer |
| Units | Square Feet |

**Full Baths**

|  |  |
| --- | --- |
| Description | The number of full baths in the house. |
| Data Type | Integer |
| Units | Count |

**Garage Year Built**

|  |  |
| --- | --- |
| Description | The year the garage was built. If it was initially built at the same time as the house, the value will be the same as the year the house was built. |
| Data Type | Integer |
| Units | Year |

**Garage Cars**

|  |  |
| --- | --- |
| Description | The number of cars the garage can house. |
| Data Type | Integer |
| Units | Count |

**Garage Area**

|  |  |
| --- | --- |
| Description | The total square feet of the garage. |
| Data Type | Integer |
| Units | Square Feet |

**House Age**

|  |  |
| --- | --- |
| Description | An engineered feature computed by subtracting the year built from the year sold. |
| Data Type | Integer |
| Units | Age in years |

**Remodel Age**

|  |  |
| --- | --- |
| Description | An engineered feature computed by subtracting the most recent remodeling or additions were added to the house. If no remodeling or additions to the house were performed, this is the same as the house age. |
| Data Type | Integer |
| Units | Age in years |

**Total Square Feet**

|  |  |
| --- | --- |
| Description | The total of all **living** areas, above and below grade, on all floors. This counts only finished area and excludes the garage, if any. |
| Data Type | Integer |
| Units | Square Feet |

**Total Square Feet Plus Garage**

|  |  |
| --- | --- |
| Description | This is an engineered feature that is computed by adding the total square feet and the garage area. |
| Data Type | Integer |
| Units | Square Feet total |

**Total Baths**

|  |  |
| --- | --- |
| Description | This is an engineered feature by adding the sum of the bathrooms, full or half, above or below grade. |
| Data Type | Integer |
| Units | Count |

**Price per Square Foot**

|  |  |
| --- | --- |
| Description | This is an engineered feature that is computed by dividing the sale price by the total square feet. |
| Data Type | Float/Real |
| Units | The price in US Dollars and cents per square foot of living area. |

**Quality X Square Feet**

|  |  |
| --- | --- |
| Description | An engineered feature that is computed from multiplying the quality (1-10) times the total living area (total square feet). |
| Data Type | Integer |
| Units | A weighted value |

**Exterior Quality**

|  |  |
| --- | --- |
| Description | The subjective quality of the exterior of the house: 10 Very Excellent  9 Excellent  8 Very Good  7 Good  6 Above Average  5 Average  4 Below Average  3 Fair  2 Poor  1 Very Poor |
| Data Type | Integer |
| Units | Rating |

**Basement Quality**

|  |  |
| --- | --- |
| Description | A subjective rating of the quality of the basement: 10 Very Excellent  9 Excellent  8 Very Good  7 Good  6 Above Average  5 Average  4 Below Average  3 Fair  2 Poor  1 Very Poor |
| Data Type | Integer |
| Units | Rating |

**Kitchen Quality**

|  |  |
| --- | --- |
| Description | A subjective rating of the quality of the kitchen: 10 Very Excellent  9 Excellent  8 Very Good  7 Good  6 Above Average  5 Average  4 Below Average  3 Fair  2 Poor  1 Very Poor |
| Data Type | Integer |
| Units | Rating |

**Fireplace Quality**

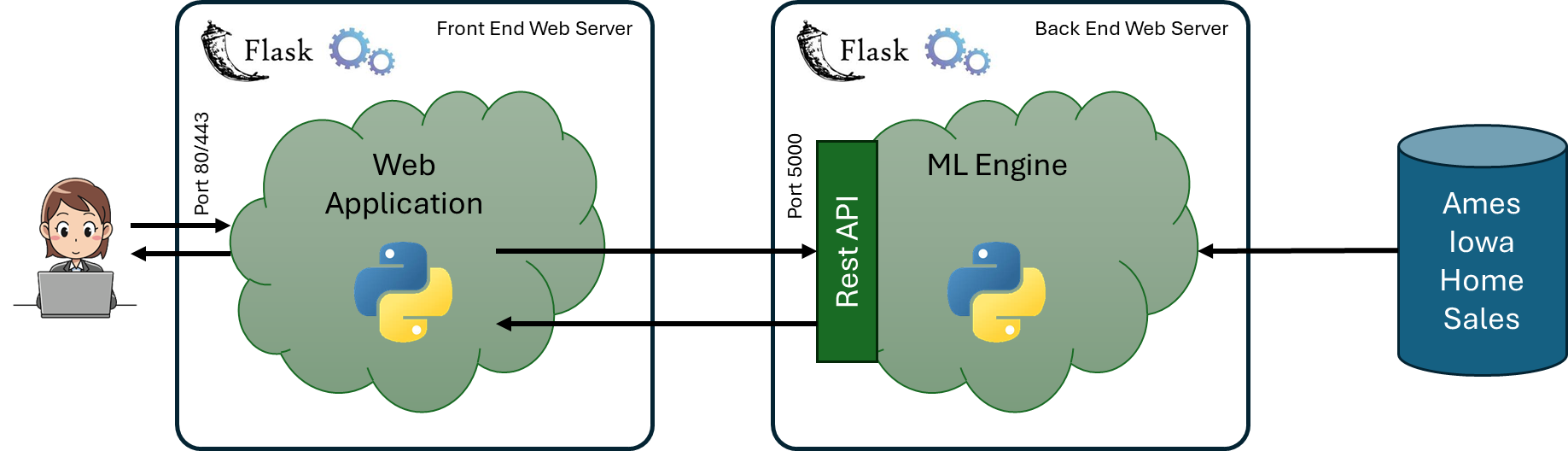
|  |  |
| --- | --- |
| Description | The subjective quality of any fireplace(s) in the house: 10 Very Excellent  9 Excellent  8 Very Good  7 Good  6 Above Average  5 Average  4 Below Average  3 Fair  2 Poor  1 Very Poor  0 No fireplaces |
| Data Type | Integer |
| Units | Rating |

# Software Architecture Document

The solution will use two separate web applications deployed in their own web servers. The primary reason for this approach is to provide an isolated runtime environment for the model engine and the model data. This provides the maximum amount of memory for it to use for maintaining the model and performing its analysis.

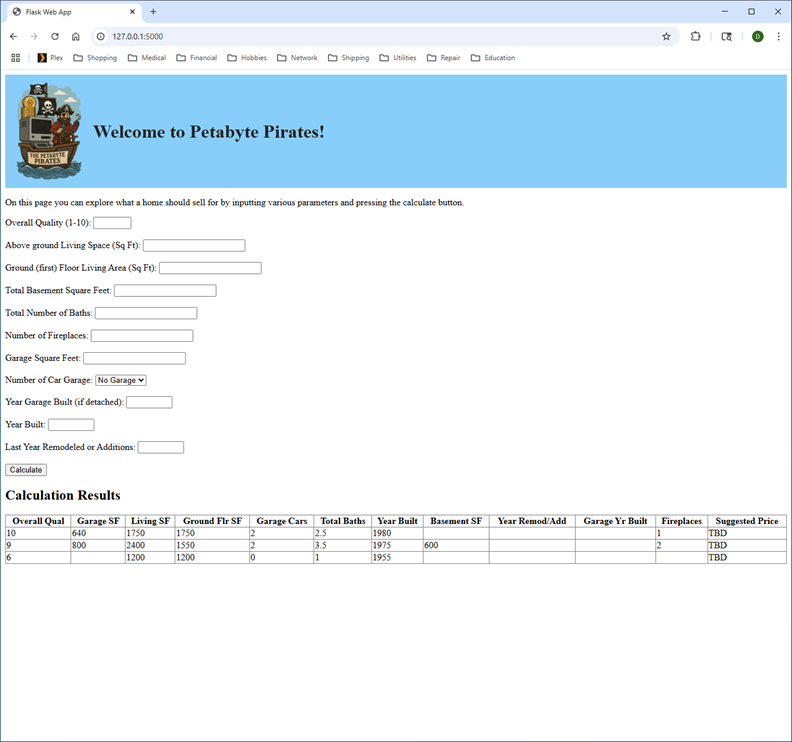
The other reason to do this is to isolate the model from the end user directly. Instead of exposing the model directly, the model will expose an API that the front-end will use to make pricing requests or other model queries. This moves the query construction, user input validation, and user interface management to the front-end web application.

This is shown in the following diagram:

This diagram shows the separate web servers, each running a separate copy of Flask. The front-end web server supports the user interface, while the back-end web server supports the model.

The user interface is intended to provide a means for the user to request pricing queries and comparisons between multiple requests.

The conceptual UI design is shown in the following diagram:



This user interface consists of a form for the user to input various characteristics which are mapped to the model features (either extracted or engineered features). The user interface validates all inputs, and if valid calls the pricing engine to predict a price given the user’s data. The result of that prediction, as well as the original data used, is retained in a table underneath the form. All columns of the results table are sortable, allowing the user to perform comparisons between multiple pricing requests.

# Model Report

A report to provide details on a specific experiment (model) - possibly one of many

## Analytic Approach

* What is the target variable?
* What kind of model was built?

## Model Description

* Models and Parameters
  + Description or images of data flow graph
    - if AzureML, link to:
      * Training experiment
      * Scoring workflow
  + What learner(s) were used?
  + Learner hyper-parameters

## Results (Model Performance)

* ROC/Lift charts, AUC, R^2, MAPE as appropriate
* Performance graphs for parameters sweeps if applicable

## Model Understanding

* Variable Importance (significance)
* Insight Derived from the Model

## Conclusion and Discussions for Next Steps

* Conclusion
* Discussion on overfitting (if applicable)
* What other Features Can Be Generated from the Current Data
* What other Relevant Data Sources Are Available to Help the Modeling

# Exit Report

This is concise document that includes an overview of the entire project, including details of each stage and learning. If a section isn't applicable (e.g. project didn't include a ML model), simply mark that section as "Not applicable". Suggested length between 5-20 pages. Code should mostly be within code repository (not in this document).

Customer: <Enter Customer Name>

Team Members: <Enter team member' names. Please also enter relevant parties names, such as team lead, Account team, Business stakeholders, etc.>

## Overview

<Executive summary of entire solution, brief non-technical overview>

## Business Domain

<Industry, business domain of customer>

## Business Problem

<Business problem and exact use case(s), why it matters>

## Data Processing

<Schema of original datasets, how data was processed, final input data schema for model>

## Modeling, Validation

<Modeling techniques used, validation results, details of how validation conducted>

## Solution Architecture

<Architecture of the solution, describe clearly whether this was actually implemented or a proposed architecture. Include diagram and relevant details for reproducing similar architecture. Include details of why this architecture was chosen versus other architectures that were considered, if relevant>

## Benefits

### Company Benefit

(internal only. Double check if you want to share this with your customer)

<What did our company gain from this engagement? ROI, revenue, etc>

### Customer Benefit

What is the benefit (ROI, savings, productivity gains etc) for the customer? If just POC, what is estimated ROI? If exact metrics are not available, why does it have impact for the customer?>

## Learnings

### Project Execution

<Learnings around the customer engagement process>

### Data science / Engineering

<Learnings related to data science/engineering, tips/tricks, etc>

### Domain

<Learnings around the business domain, >

### Product

<Learnings around the products and services utilized in the solution >

### What's unique about this project, specific challenges

<Specific issues or setup, unique things, specific challenges that had to be addressed during the engagement and how that was accomplished>

### Links

<Links to published case studies, etc.; Link to git repository where all code sits>

### Next Steps

<Next steps. These should include milestones for follow-ups and who 'owns' this action. E.g. Post- Proof of Concept check-in on status on 12/1/2016 by X, monthly check-in meeting by Y, etc.>

# Appendix

<Other material that seems relevant – try to keep non-appendix to <20 pages but more details can be included in appendix if needed>