This report was written under a strict deadline during a normal workweek as a pre-assignment for a job interview, hence the report can feel incomplete nor not finalized at times. However, it covers all the key aspects of a data science project.

The underlying dataset cannot be shared unfortunately.

**Business Context / Introduction**

Goal is to predict the “Outcome” based on the variables X1 to X10.

The problem seems like a binary classification task because the values of the outcome varies between 0 and 1.

Model needs to be used in production and real-life, hence the model needs to generalize in the best possible way to unseen data.

The overall process goes as follow:

1. Data
   1. Understanding
      1. How is the data distributed?
      2. Are there segments within the data that stand out?
      3. Are there missing values? If there are how to deal with them (median, mode, average estimates or remove rows altogether)?
      4. How is the outcome variable distributed? Are we predicting rare classes or is the dataset balanced?
      5. How skewed the data is?
      6. Any outliers?
   2. Preparation
      1. Treat missing values appropriately.
      2. Remove outliers (if needed).
      3. Scale (and standardize if needed).
      4. Transform into the final form for model building, cross validation and grid search.
2. Modeling
   1. Set aside 20% of the data for test set to estimate in the end how well the produced model generalizes to unseen data.
   2. Try different classification models.
   3. Search the hyperparameter space using Grid Search .

**Understanding and preparing the data**

Total sample count is 10,000.

Using pandas “describe” function we get the following stats for non-categorical variables. 25%, 50% and 75% are the percentiles of the dataset.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Outcome** | **X1** | **X2** | **X3** | **X4** | **X5** | **X6** | **X7** | **X9** |
| count | 10000 | 10000 | 10000 | 10000 | 10000 | 9999 | 10000 | **10000** | 10000 |
| mean | 0.505 | 0.496 | 49.666 | 0.503 | 0.500 | 0.498 | 0.498 | **1.994** | 0.497 |
| std | 0.500 | 0.339 | 28.908 | 0.289 | 0.288 | 0.240 | 0.221 | **0.771** | 0.203 |
| min | 0.000 | -17.300 | 0.010 | 0.000 | 0.000 | 0.011 | 0.003 | **1.000** | 0.004 |
| 25% | 0.000 | 0.248 | 24.302 | 0.254 | 0.251 | 0.296 | 0.322 | **1.000** | 0.351 |
| 50% | 1.000 | 0.496 | 49.493 | 0.504 | 0.500 | 0.498 | 0.497 | **2.000** | 0.496 |
| 75% | 1.000 | 0.748 | 74.517 | 0.754 | 0.750 | 0.700 | 0.673 | **3.000** | 0.642 |
| max | 1.000 | 1.000 | 99.982 | 1.000 | 1.000 | 0.993 | 0.989 | **3.000** | 0.994 |

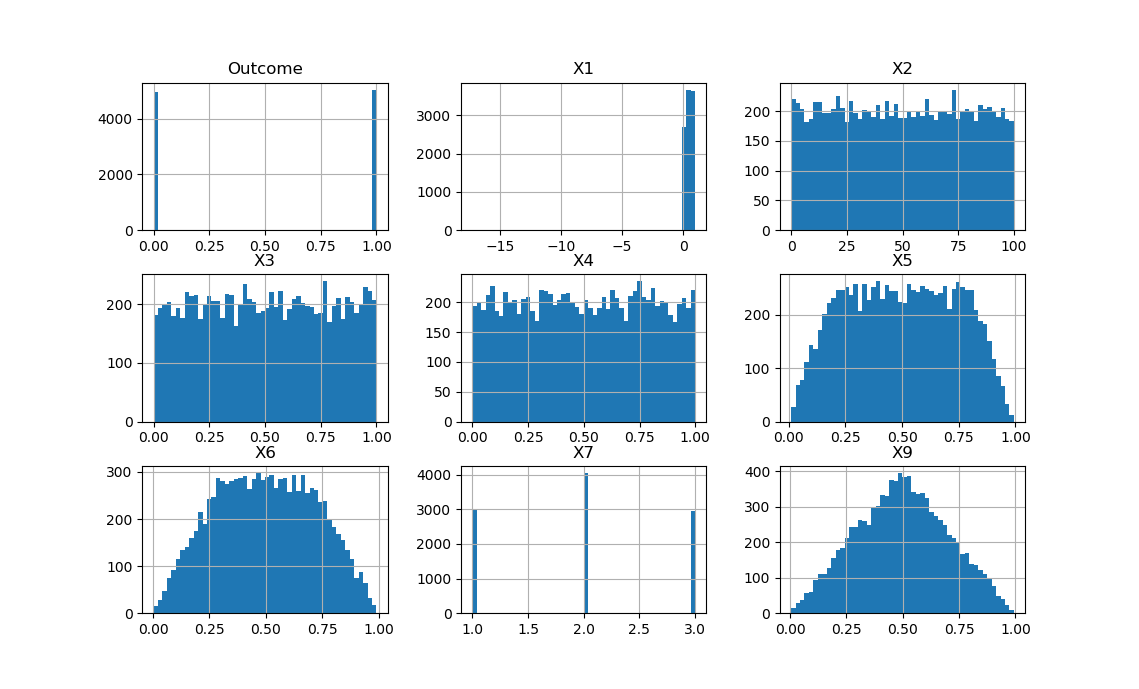
From the above table we can see that X1, X2, X3, X4 are quite similarly distributed, except that X2 is between 0-100 while the other ones are generally between 0 and 1, except for a potential outlier for X1 with the minimum value of -17.3 that is a bit suspicious. We will get back to this outlier in a bit. Also, important to note that given the scales are different this can impact the performance of the training algorithm, hence we want to transform all the continuous variables to be between 0-1. This will be done at a later stage.

X5 also seems to be missing a value and with a detailed analysis there indeed is a row with missing value. Seems like that this is the only missing value in the whole dataset. Given this represents only 1/10,000 of the whole dataset we can safely exclude it from further analysis instead of using other methods to guess its value.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Outcome** | **X1** | **X2** | **X3** | **X4** | **X5** | **X6** | **X7** | **X8** | **X9** | **X10** |
| 1 | 0.427281 | 0.185748 | 0.973967 | 0.249286 | NA | 0.094722 | 2 | A | 0.214569 | B |

**X7** looks like a categorical value with number representation (1, 2, 3).

Next let’s look at the histograms for each numerical independent variables.



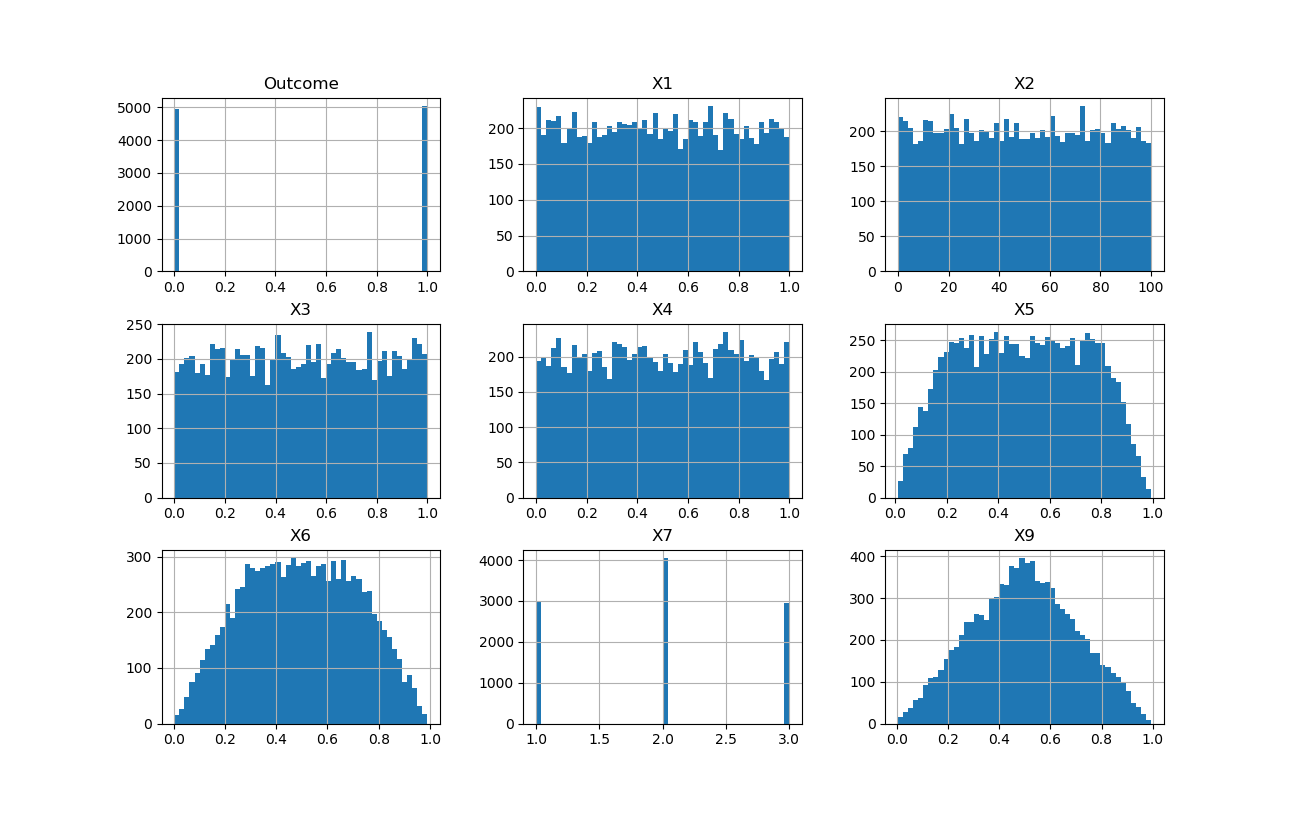
First, the suspicion about the X1 = -17.3 being an outlier is confirmed by looking at the above graph: it clearly is an outlier / a false data and can be safely removed from the dataset.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Outcome** | **X1** | **X2** | **X3** | **X4** | **X5** | **X6** | **X7** | **X8** | **X9** | **X10** |
| 1 | -17.3 | 41.0715 | 0.696229 | 0.386812 | 0.199891 | 0.360762 | 1 | B | 0.249922 | B |

Also, important to note that the target class is well balanced between 0 and 1 classes, hence we don’t need to take steps such that one would take in case of highly skewed datasets (e.g. rare class prediction).

Lastly, important to note that our suspicion about **X7** being a categorical variable is confirmed.

Here’s how the distributions look like after dropping the missing value and the outlier:



Next let’s analyze the categorical variables (X7, X8, X10).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **X7** | **Count** |  | **X8** | **Count** |  | **X10** | **Count** |
| 2 | 4052 |  | B | 3950 |  | B | 5043 |
| 1 | 3001 |  | A | 3059 |  | A | 4955 |
| 3 | 2945 |  | C | 2989 |  |  |  |

The variable X10 category B seems to be suspiciously close to the number of positive labels in the dataset. And as suspected, it turns out that X10 is actually the outcome variable (B = 1 and A = 0), hence it can be dropped from further analysis. The problem is now to predict the outcome variable based on X1-X9. Nothing else stands out from analyzing the categories.

Next, let’s analyze the correlations all-up and by segment. Coloring for correlations with itself has been removed to help spot the important information.

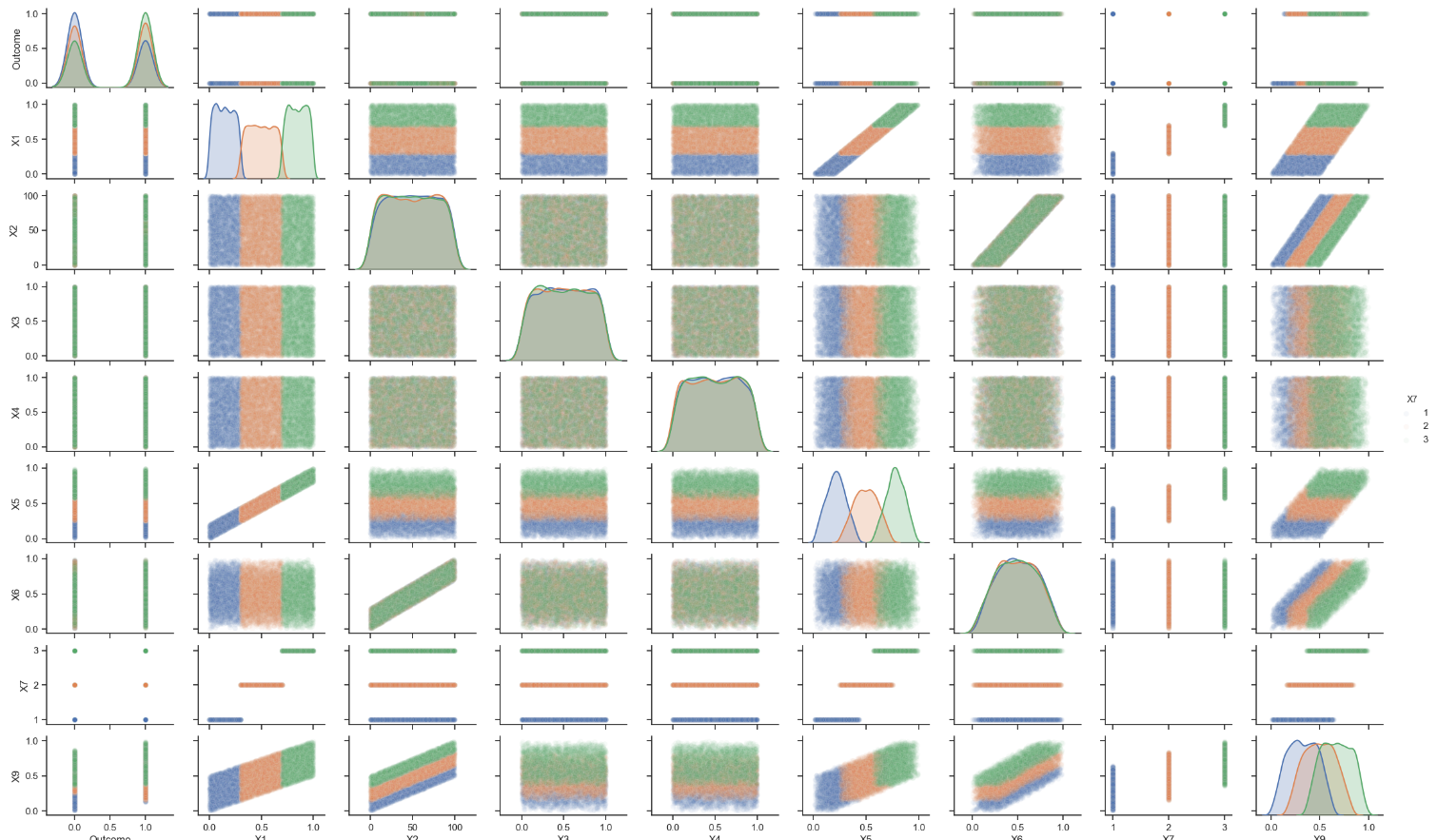
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Outcome | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X9 |
| Outcome | 1.00 | 0.14 | -0.01 | 0.00 | 0.00 | 0.14 | -0.01 | 0.18 | 0.09 |
| X1 | 0.14 | 1.00 | -0.01 | -0.01 | 0.01 | 0.97 | -0.02 | 0.94 | 0.70 |
| X2 | -0.01 | -0.01 | 1.00 | 0.00 | 0.00 | -0.01 | 0.92 | -0.02 | 0.70 |
| X3 | 0.00 | -0.01 | 0.00 | 1.00 | 0.00 | -0.01 | 0.00 | -0.01 | -0.01 |
| X4 | 0.00 | 0.01 | 0.00 | 0.00 | 1.00 | 0.00 | -0.01 | 0.00 | 0.00 |
| X5 | 0.14 | 0.97 | -0.01 | -0.01 | 0.00 | 1.00 | -0.01 | 0.91 | 0.68 |
| X6 | -0.01 | -0.02 | 0.92 | 0.00 | -0.01 | -0.01 | 1.00 | -0.02 | 0.64 |
| X7 | 0.18 | 0.94 | -0.02 | -0.01 | 0.00 | 0.91 | -0.02 | 1.00 | 0.66 |
| X9 | 0.09 | 0.70 | 0.70 | -0.01 | 0.00 | 0.68 | 0.64 | 0.66 | 1.00 |

Key findings:

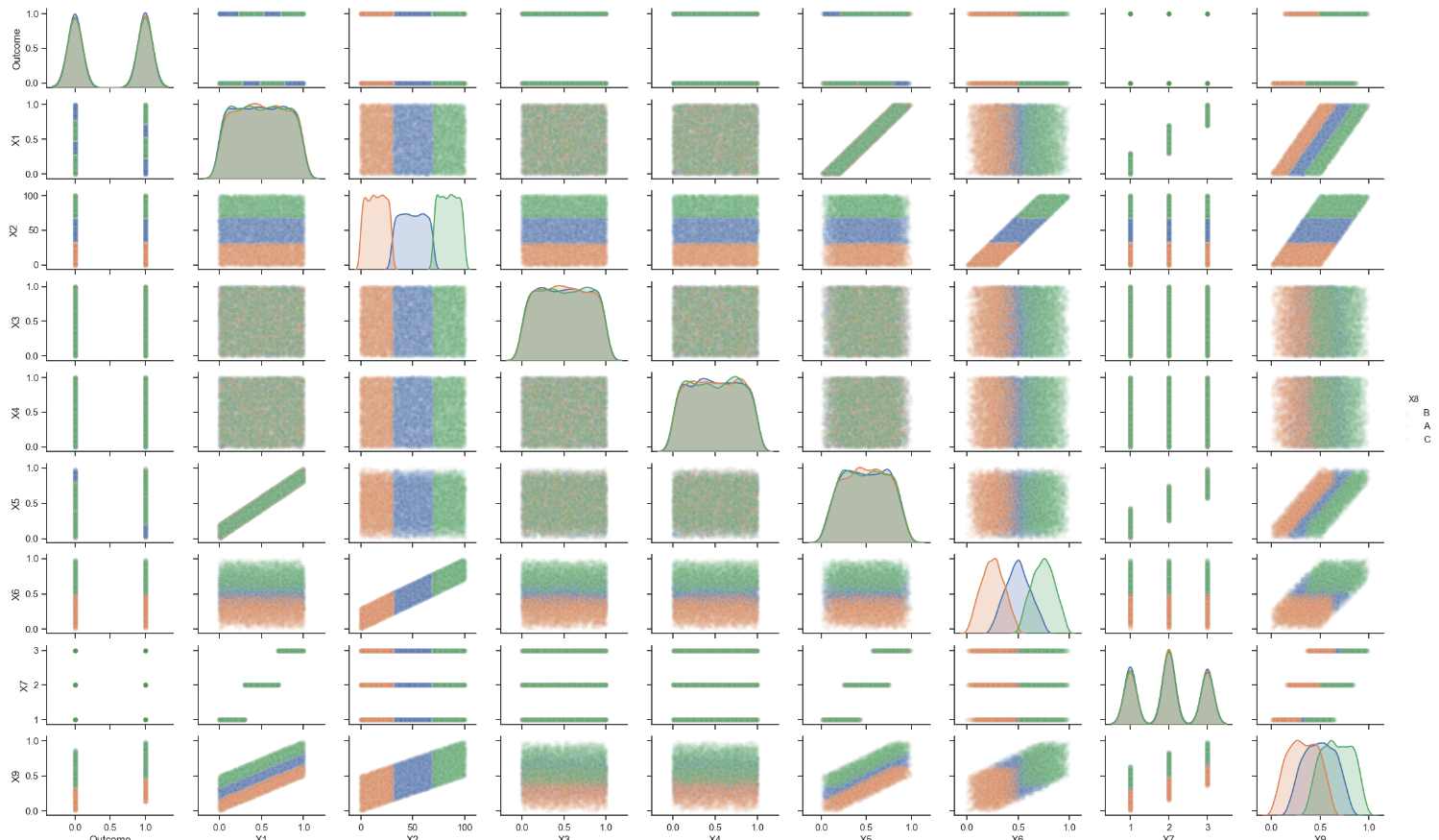
* X1 is highly correlated with X5, X7 and X9.
* X2 is highly correlated with X6 and X9.
* X5 is highly correlated with X1, X7 and X9.
* X6 is highly correlated with X2 and X9.
* X7 is highly correlated with X2, X5 and X9.
* X9 is correlated with X2, X3, X5, X6, X7
* None of the independent variables are really correlated with the outcome i.e. dependent variable.

Next we’ll look at how the data looks like by category defined by X7 (1,2,3) and X8 (A, B, C) in a scatter matrix format. We will use Seaborn pairplot.

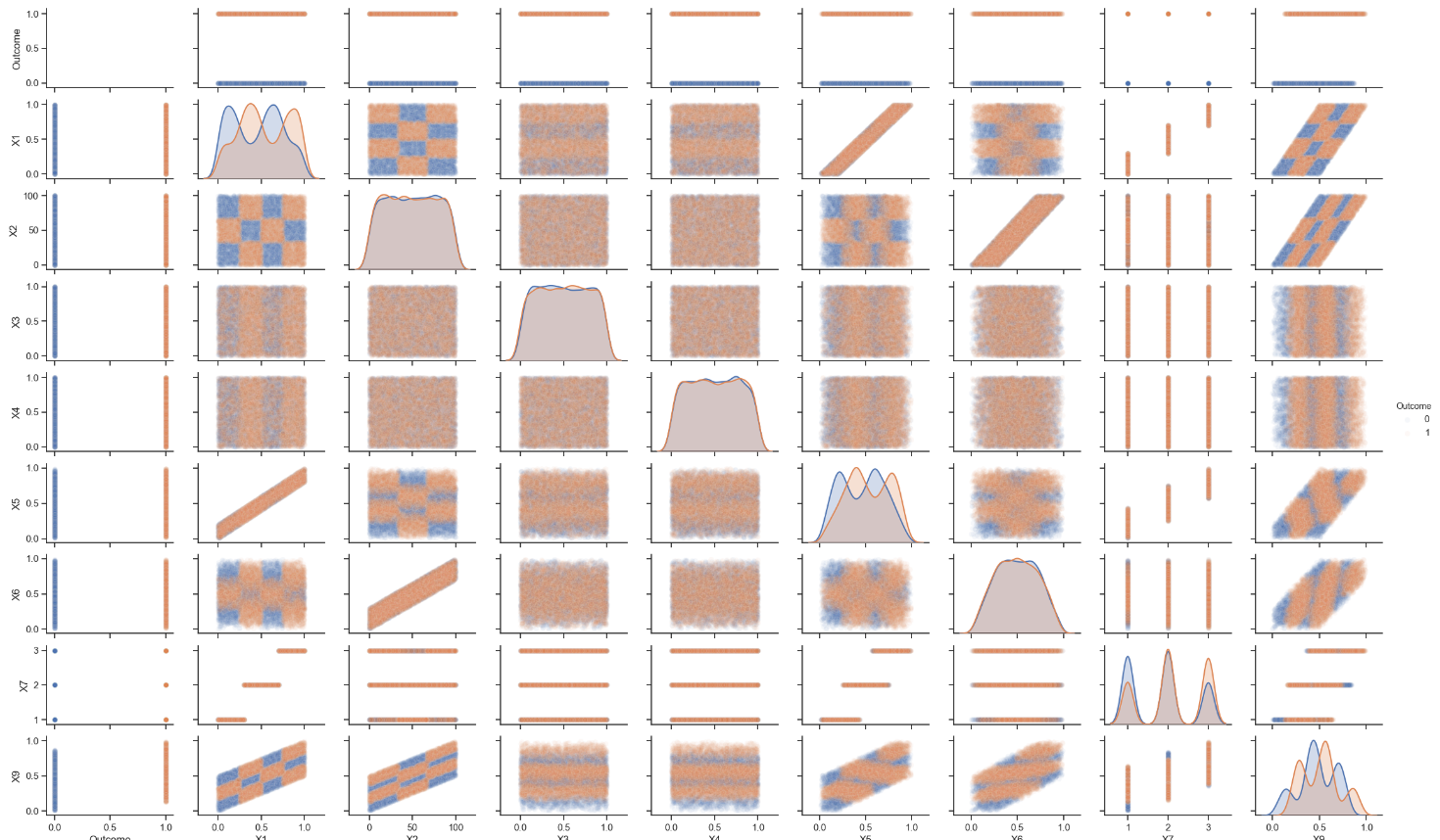
Data segmented by X7:



Data segmented by X8:



Data segmented by class:



Couple of thoughts per above graphs:

* **By segmenting the data with X7 and X8 it can be seen that X1 and X2 will explain the outcome variable, assuming the data truly is separable given the limitations in the accuracy of the above visualizations.**
* Decision trees will probably work well because there are clear clusters and hence the data seems separable in specific areas.

For the sake of completeness, we’ll evaluate other models and feature importance as well even with the above findings.

**Modeling**

Before beginning with the modeling exercise, we will set aside 20% of the data for test set. The purpose of test set is to estimate how well our model generalizes to unseen data so we can asses its performance before using it in live / production environment.

We are not going to try combine or derive new variables of the existing ones because we don’t know the meaning of them.

AUC is selected to be the KPI for model selection. AUC describes the overall quality of the model by estimating the area under the receiving operating characteristics curve. The ROC displays false-positive-rate (x) vs. true-positive-rate (y) at different thresholds. 0.5 equals random guessing while 1.0 is a perfect classifier.

Feature selection:

ExtraTreesClassifier from scikit is used to train randomized decision trees for estimating feature importance. The model is built for the full training set without validation because the purpose is to get a sense of the important features. The model actually yields AUC score of 1.0 potentially indicating severe overfitting, or then the model just works as assumed earlier. Despite the potential overfitting, we can still use the feature importances to get an understanding what is important and what is not.

Feature importances:

|  |  |
| --- | --- |
| **Feature** | **Importance** |
| X1 | 0.283 |
| X9 | 0.141 |
| X2 | 0.125 |
| X8\_B | 0.097 |
| X5 | 0.084 |
| X8\_C | 0.052 |
| X6 | 0.043 |
| X8\_A | 0.038 |
| X4 | 0.034 |
| X3 | 0.032 |
| X7\_2 | 0.026 |
| X7\_3 | 0.025 |
| X7\_1 | 0.020 |

We could consider dropping the X7, X3, X4 and X6 from the final model altogether given they are not potentially too important, but let’s try first with all features.

The modeling process goes as follows:

1. Build a hyperparameter set for evaluation for each model.
2. Find the best hyperparameter in hyperparameter space using cross validation with 5 folds on the training set.
3. Build a new model using the best hyperparameter using the whole training dataset.
4. Evaluate the training AUC.
5. Estimate the generalization error by evaluating AUC for the test set.

**Results**

SVM:

SVC\_params = [

{'kernel':['linear'], 'C':[1, 10, 100, 1000]},

{'kernel':['rbf'], 'C':[1, 10, 100, 1000], 'gamma': [0.001, 0.0001]},

]

Best parameters: {'C': 1000, 'gamma': 0.001, 'kernel': 'rbf'}

**AUC training: 0.733**

Logistic regression:

Logistic\_params = [

{'tol':[1, 10, 25], 'penalty':['l1', 'l2']},

]

Best parameters: {'penalty': 'l1', 'tol': 1}

**AUC training: 0.597**

Random Forest:

RandomForest\_params = [

{ 'n\_estimators': [10,200,500],

'bootstrap': [True],

'max\_depth': [10, 50, 100, None],

'max\_features': ['auto'],

'min\_samples\_leaf': [2, 4],

'min\_samples\_split': [2, 10],

},

]

**AUC training: 0.999**

Such a high AUC for a training set can indicate major overfitting. Let’s see what we get for the test set.

**Test set AUC: 0.996**

**Hence we can conclude the model is not overfitting at all and seems to generalize well to unseen data**. Let’s just take a final look on the confusion matrix:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Predicted | |
|  |  | 0 | 1 |
| True | 0 | 979 | 3 |
|  | 1 | 3 | 1015 |

**In conclusion, the random forest fits the data almost perfectly as expected and it generalizes well into unseen data. Same result will be achieved with all features and only using X1 and X2.**