Shaun Markham 09/03/2020

Hastings Direct Data Science Assessment

Section 1: Outline and Explanation

The problem at hand was defined as a binary classification challenge, with the main aim being to predict if the sale of an insurance policy is made given individual customer information. The provided data can be described as follows.

Feature Name	Type	Comments
Age	Continuous	The age of the customer
Veh_Value	Continuous	The market value of the car in GBP
Tax	Continuous	An additional fee on top of the price, insurance premium tax rate
Price	Continuous	The annual price to be paid for the policy in question
Veh_Mileage	Continuous	The mileage of the vehicle in question
Credit_Score	Continuous	The customer's credit score
License_Length	Continuous	The length of time a customer has held their license in years
Date	Date	The start date for the policy
Marital_Status	Categorical	The customer's marital status
Payment_Type	Categorical	The method of payment
Veh_Reg_Year	Categorical	The year when the vehicle was registered with the DVLA
Sale	Categorical (Target)	Whether or not the policy was sold

The method of approach chosen was one that follows the subsequent pattern:

- An initial exploratory data analysis that dealt with observing the relationships between the data. This was done to see if any intrinsic relationships could be taken advantage of so as to better the final model.
- After this the missing values seen within the data were looked at and attempted to be filled.
- Artificially created features were then added to see their effect on the model's accuracy.
- Encoding was then carried out on the required features.
- The modelling stage was then done, together with the application of different accuracy measures to grade said model's performance.
- Hyperparameter tuning was finally carried out to unlock the full potential of the model on the data at hand.

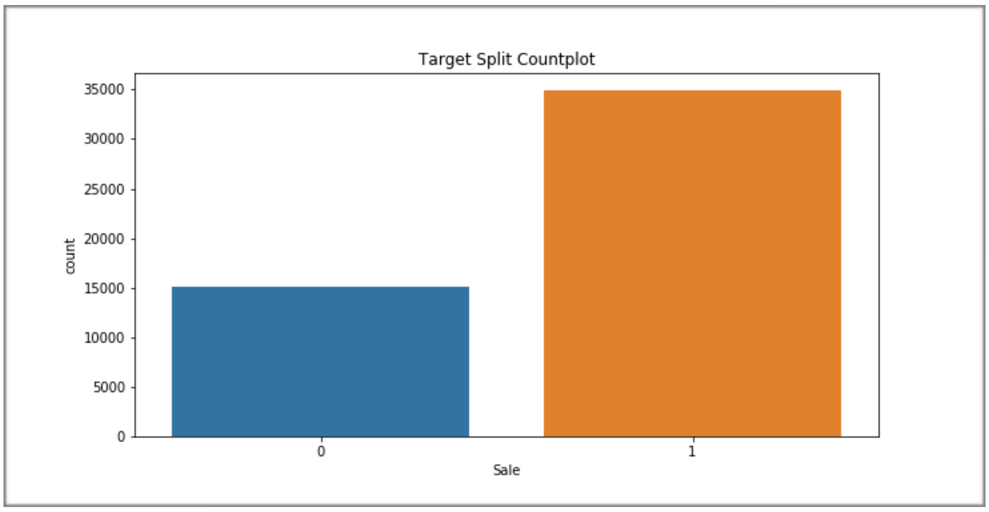
Initial Data Exploration

Before the exploration of data was carried out, an initial examination of the data was done using an *explore* method designed to report the number of missing values seen for each feature. This produced the following results:

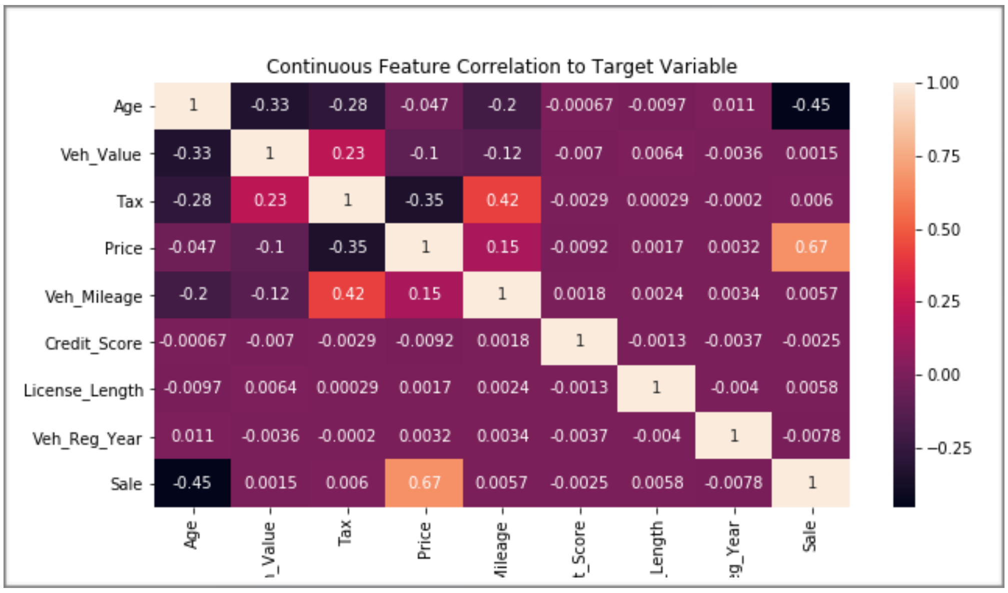
Feature Name	Data Type	Number Missing	Total Observed	Missing as % of Total
Age	Float64	483	49517	0.966
Veh_Value	Float64	525	49475	1.050
Tax	Float64	517	49483	1.034
Price	Float64	515	49485	1.030
Veh_Mileage	Float64	509	49491	1.018
Credit_Score	Float64	497	49503	0.994
License_Length	Float64	498	49502	0.996
Date	Object	497	49503	0.994
Marital_Status	Object	483	49517	0.966
Payment_Type	Object	484	49516	0.968
Veh_Reg_Year	Float64	488	49512	0.976
Sale	Int64	0	50000	0.000

This table shows us how although there is missing data throughout each feature, they should not constitute enough of a portion of the overall such that they have a negative affect on model performance. Nonetheless, each feature will be tested and an attempt will be made to impute these missing values.

After this the target class split was charted, so as to check for class imbalances. This has a hand in making a decision on which model to use, as some models are more resilient to imbalanced data.



For the purpose of data exploration, the relationships of features with the target were tested. Initially, a correlation plot was carried out so as to identify which features are mostly related to the target. After that each individual feature was examined and plotted to check for any trends that could be exploited.

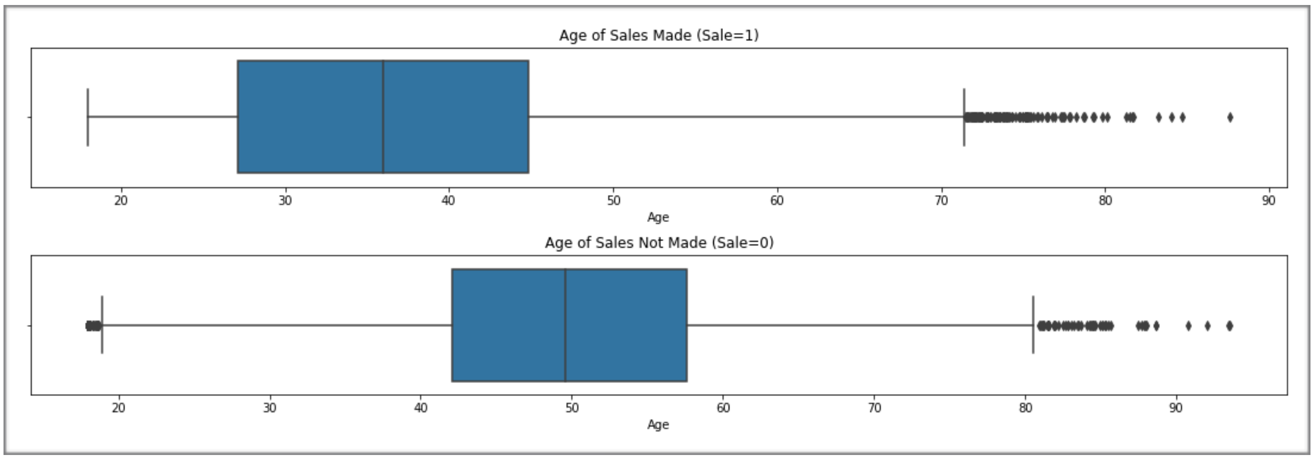


The figure shows us how the majority of features have nearly no relationship with the Sale target, with Price being the only feature that is significantly correlated.

Individual Feature Examination

Age

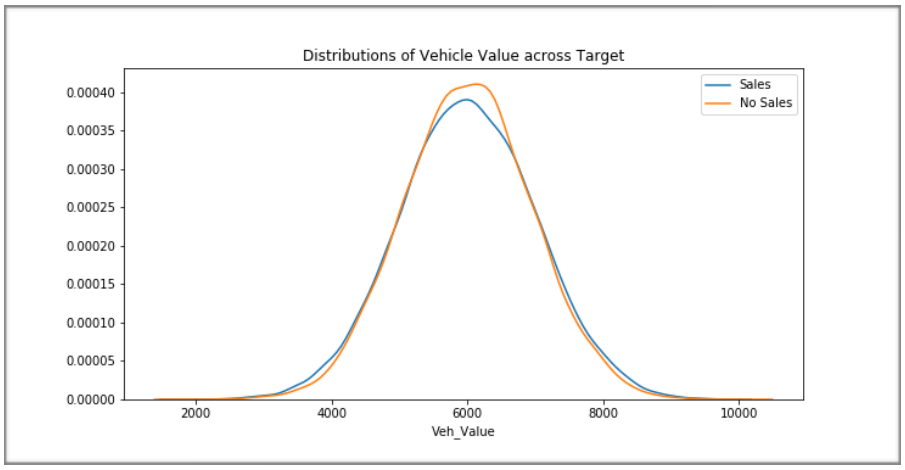
The first feature to be observed was the Age feature. The first figure seen below shows the variation of Age with respect to the target.



The box plots show the difference in dispersion across Ages. Interestingly, the median age for each group is quite different, with most offers resulting in no Sale coming from clients aged around 50, while offers taken up come from clients with a median age of 35.

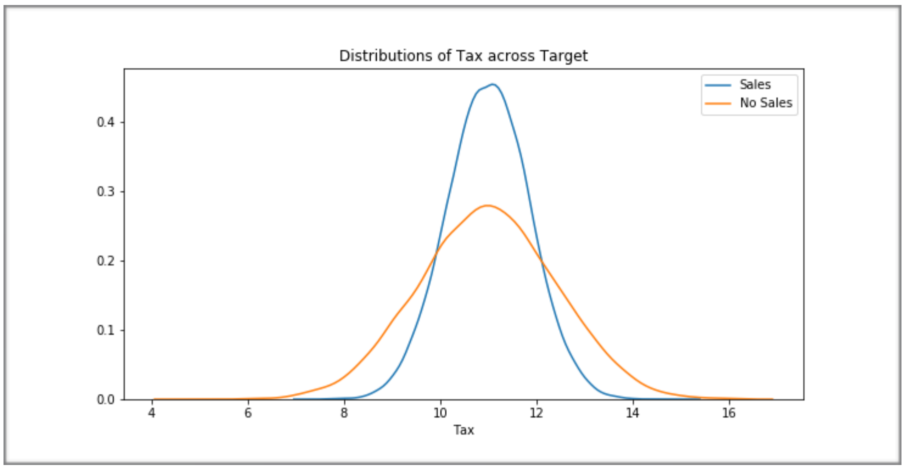
Now distributions of other features were plotted, again to see if there might be any underlying relationship that can be exploited.

Vehicle_Value



Here we can see how the distributions across the target variable are relatively similar. This can also be seen from Pearson correlation coefficient between this value and the target (-0.043).

Tax

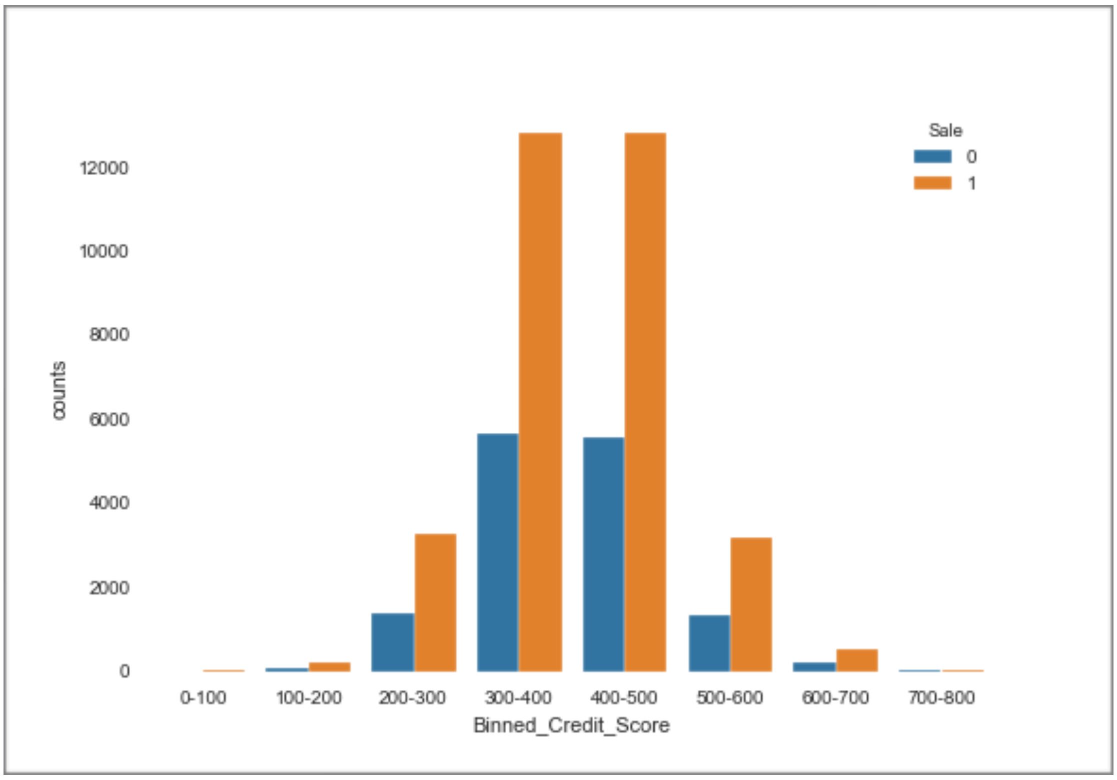


Although the distributions of Tax across the target variable differ, the correlation shown to the target is still nearly 0 (-0.045). This indicates nearly no relationship between this feature and the target.

Credit_Score

An interesting observation was seen when checking the distributions of the Credit_Score feature. Although this feature also showed nearly no relationship with the target, there were multiple instances of customers having a credit score of 9999. This was assumed to be a mistake, with these being changed to 700, the maximum achievable credit score (according to the Equifax) (the median was observed to be 500).
The customers having a credit score of 9999 were observed to check if the data that they represented had different distributions to the rest of the dataset, but this was quickly deemed as false upon comparing visually.

Another method tried to check if there was some underlying information within the Credit_Score feature was to bin the values arbitrarily (taking 100 large increments). This was then viewed visually by grouping the data according to the resulting bins across the target variable, the result of which can be seen in the following plot.



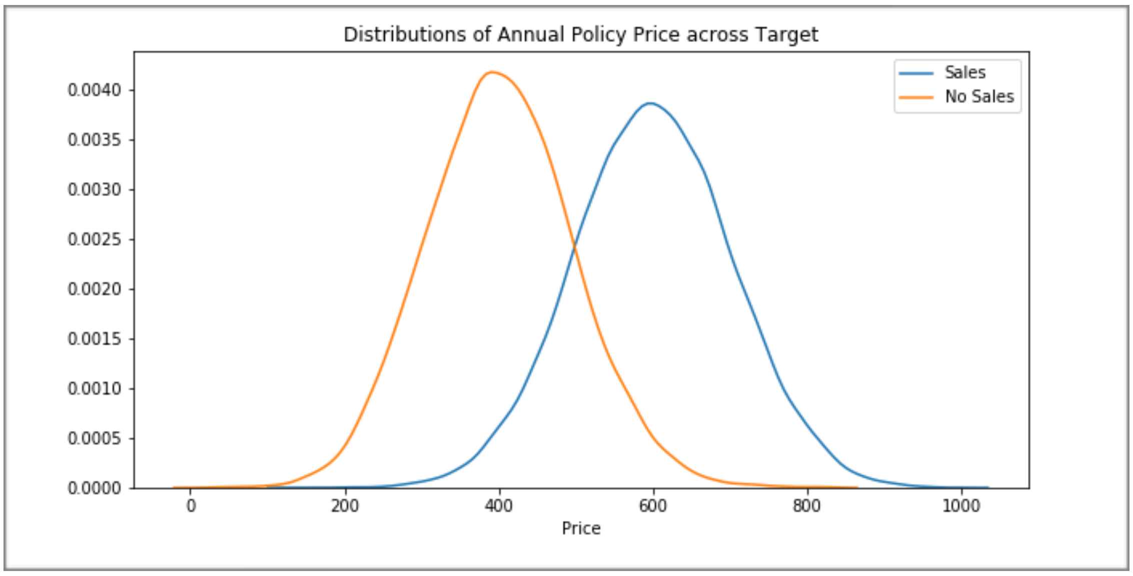
This shows how the majority of customers have a credit score between 300 and 500. It also shows how although the number of occurrences differ across credit scores, the proportion of sales to no sales are relatively similar.

License_Length

The License Length feature was then checked, with an interesting observation being made almost immediately. A number of 18 year-old customers were found to have license lengths in excess of 1 year while this feature is defined as being the number of years that a client has held a full driving license. It was decided that these rows were to be removed. A very low correlation score (0.004) was observed.

Price

The Price feature was immediately found to be the most highly correlated to the target (0.7), with the following distributions being observed.



This plot shows how the price feature is highly correlated to the target. Interestingly, the majority of sales are seen to be had at an above average price.

Veh_Mileage

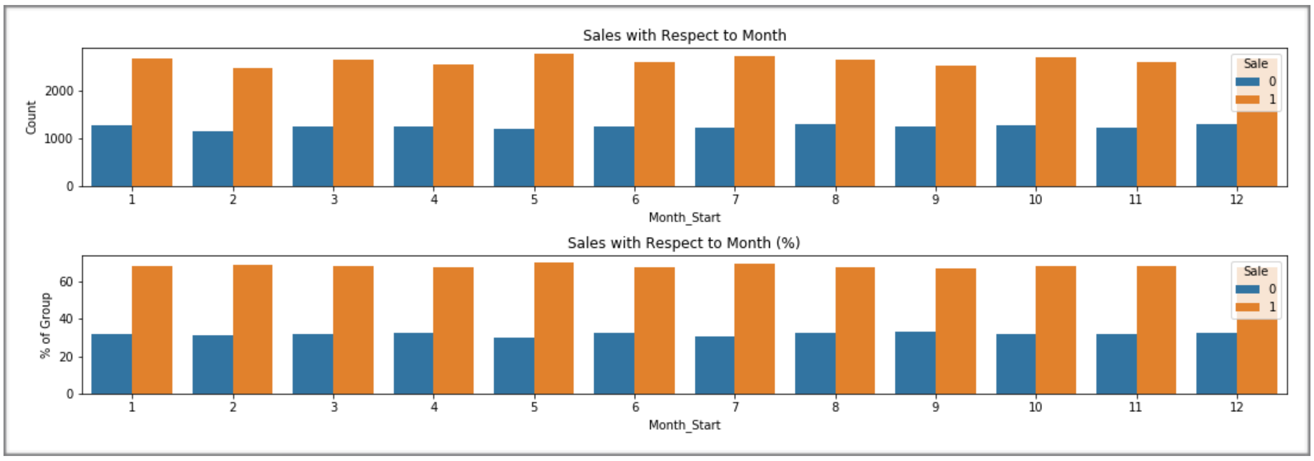
The vehicle mileage feature was found to have nearly no correlation to the target (-0.023), with the distributions closely resembling those seen previously.

Date

The date feature was then examined, but before being able to do so was split into separate columns so as to extract the maximal amount of information from it. From the provided date string, and using pandas *to_datetime()* method, 4 new features were created, namely:

- A Year feature (*Year_Start*)
- A Month feature (*Month_Start*)
- A Day of Month feature (*Day_Start*)
- A Day of Week feature (*DayOfWeek_Start*)

Upon doing this transformation, the newly added Year feature was found to contain only one value (2016), and was therefore dropped. The month feature can be seen represented as a bar chart below.



Here a chi-squared test was used to check the relationship with the target variable, due to its categorical nature. More specifically, the test used was *scipy*'s *chi2_contingency*, which returns a p-value. With a p-value less than 0.05, we can reject the null hypothesis, in this case being independence between the features in question. Here we are using this to check the dependence of this feature with the target, therefore a value of 0 would show a strong relationship and therefore would conclude that we can reject the null hypothesis.

In this case the test produced a p-value of 0.175, indicating no significance. The same was done for each newly added feature, with *DayOfWeek_Start* having the best value (0.072), but still indicating no significance.

Marital_Status

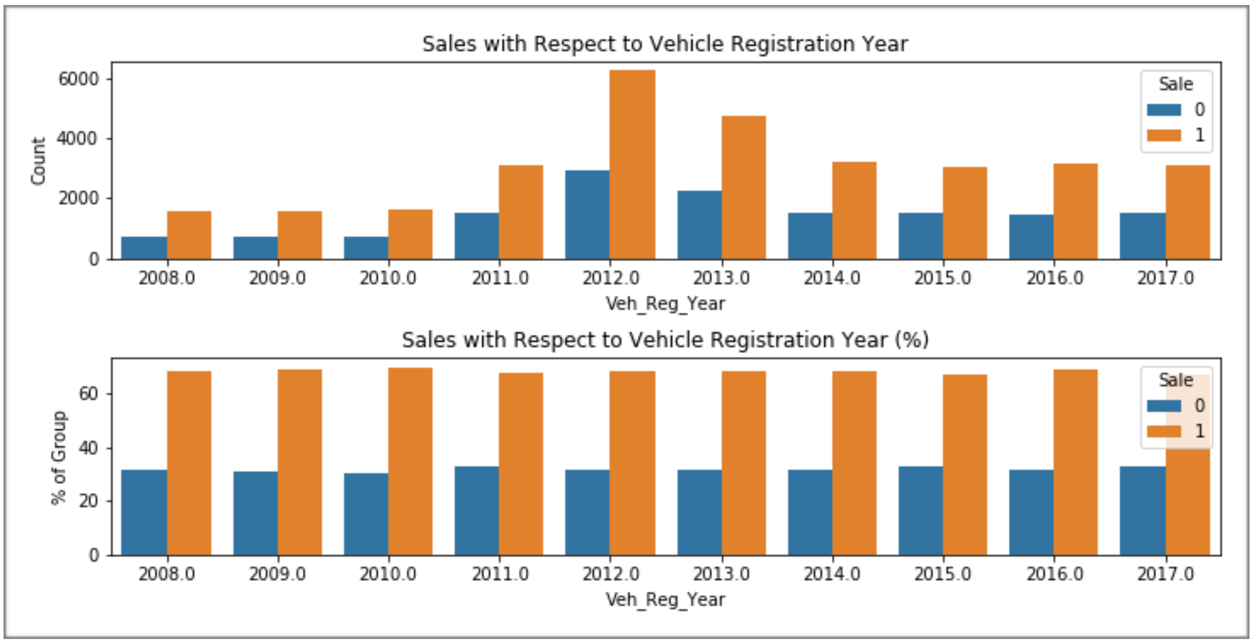
The same bar plot was carried out on the marital status feature. Interestingly, this showed how the majority of customers in the data are married. The chi-squared test was also carried out, producing a result of 0.868, indicating no significance.

Payment_Type

With respect to the product type feature, the data was seen to contain a near 50-50 split between payments by card and by instalments. On further examination, they also contained a near identical split with respect to the target variable. The chi-squared test produced a p-value of 0.933, indicating no significance.

Veh_Reg_Year

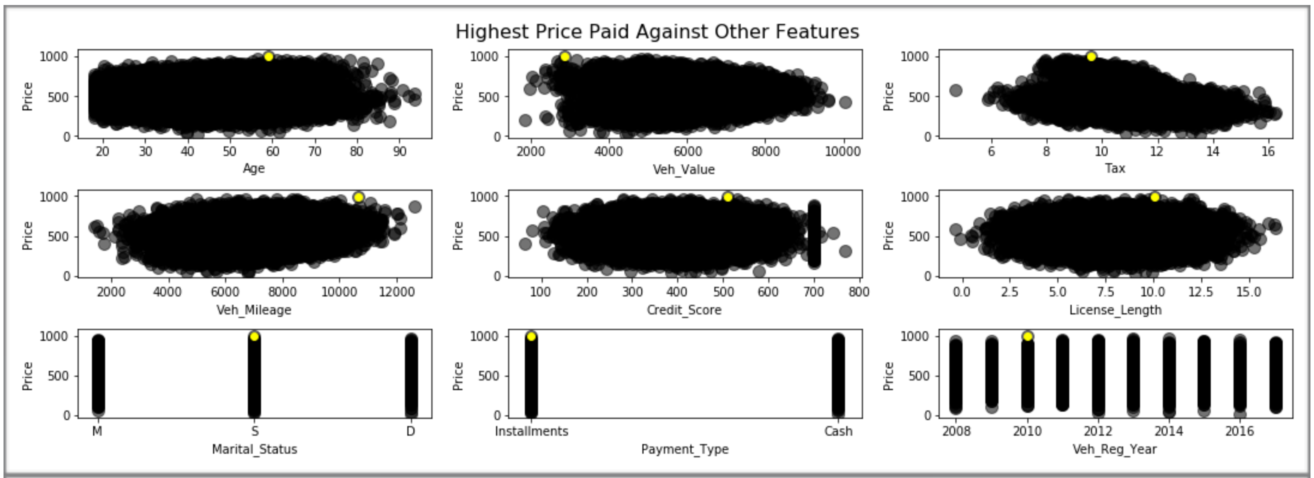
The vehicle registration year feature was also plotted as seen previously, showing a different number of registrations across years, but with each year having nearly the same ratio of sales to no sales. This can be seen in the following plot.

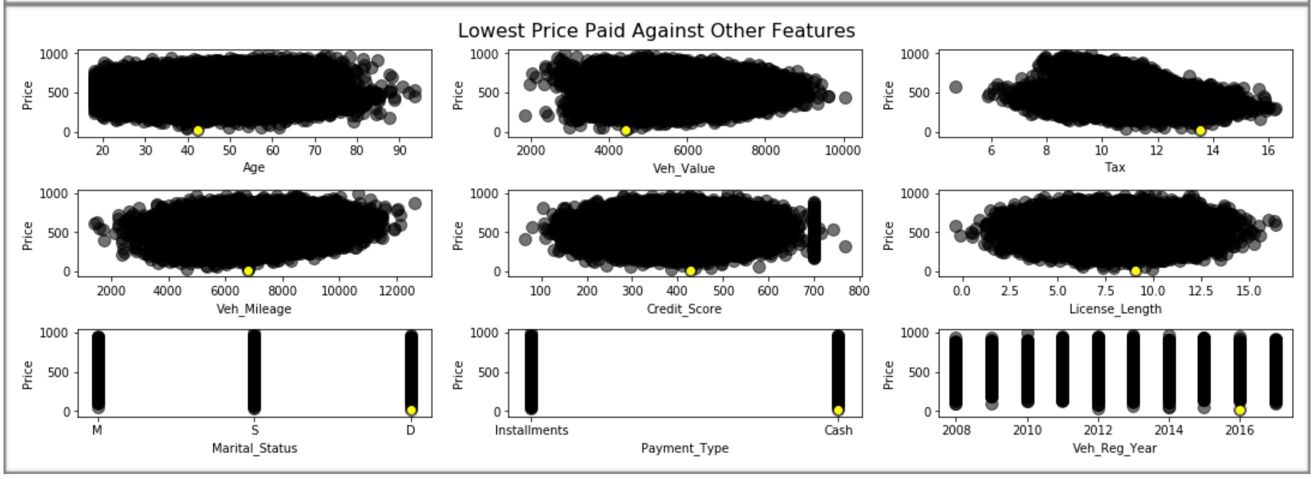


This feature was also tested using the chi-squared method, producing a p-value of 0.484, again showing no significance.

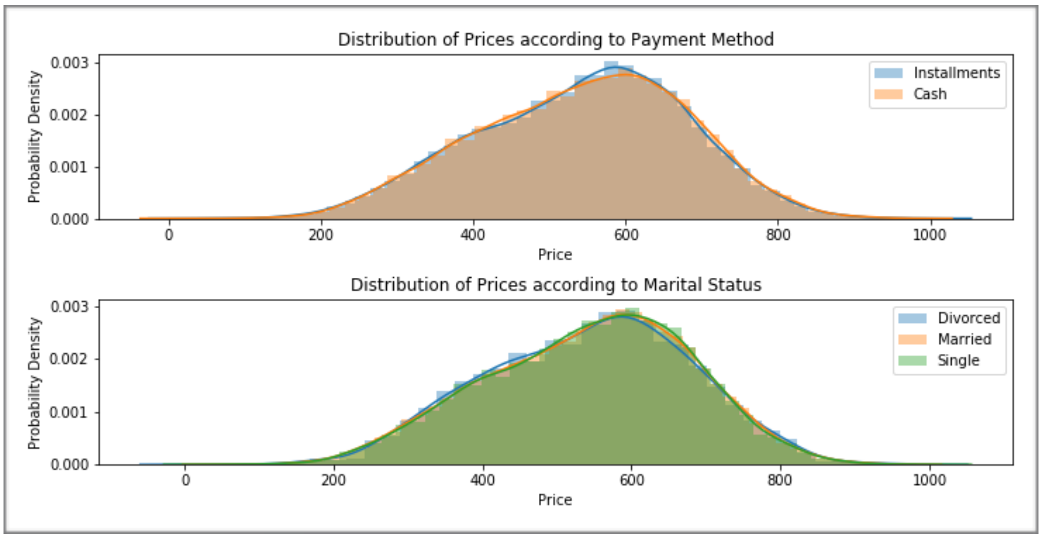
Revisiting Price

Due to the Price feature having the highest correlation to the target variable, it was revisited and taken another look at. The rows containing the minimum and maximum prices observed were plotted across all other features, so as to further visually inspect the data. These plots can be viewed below, with the points in question being highlighted in yellow and each feature being plotted against Price on the y-axis.





Another distribution plot of the price feature was created, but this time taking into consideration the different categories in *Marital_Status* and *Payment_Type.*



Which showed that for either feature, no group is particularly favoured or gets a better/worse price.

Model Selection & Evaluation Metrics

When deciding on which type of model is to be selected, several different criteria must be taken into consideration. The type of model picked here was of the gradient boosting type which are known to perform well when used on tabular data, with many of the top entries for kaggle competitions using these types of models. The most recent and successful implementations of these models are XGBoost, CatBoost and Microsoft's LightGBM. Due to LightGBM's speed of training, it was picked as the model of choice. Also, although there is a slight class imbalance within this dataset, gradient boosting machines have been shown to perform especially well in imbalanced class situations.

Model evaluation was done using multiple different evaluation measures. These were:

- The accuracy metric, which scores predictions based on if the predicted label exactly matches the corresponding set of actual targets.
- The area under the receiver operating characteristic score (roc_auc), which is a measure of how well the model can distinguish between the target groups.
- The F1 score, which is a blend between precision and recall.
- A confusion matrix.

Section 2: Model Building and Results

Imputing Missing Values

As seen in the second table of this document, the amount of missing values seen in each feature constituted a very small portion of the overall. Nonetheless, attempts were made to adjust and impute all missing values. Three different methods were attempted to impute missing values, with these being described as follows.

Imputing via GroupBy

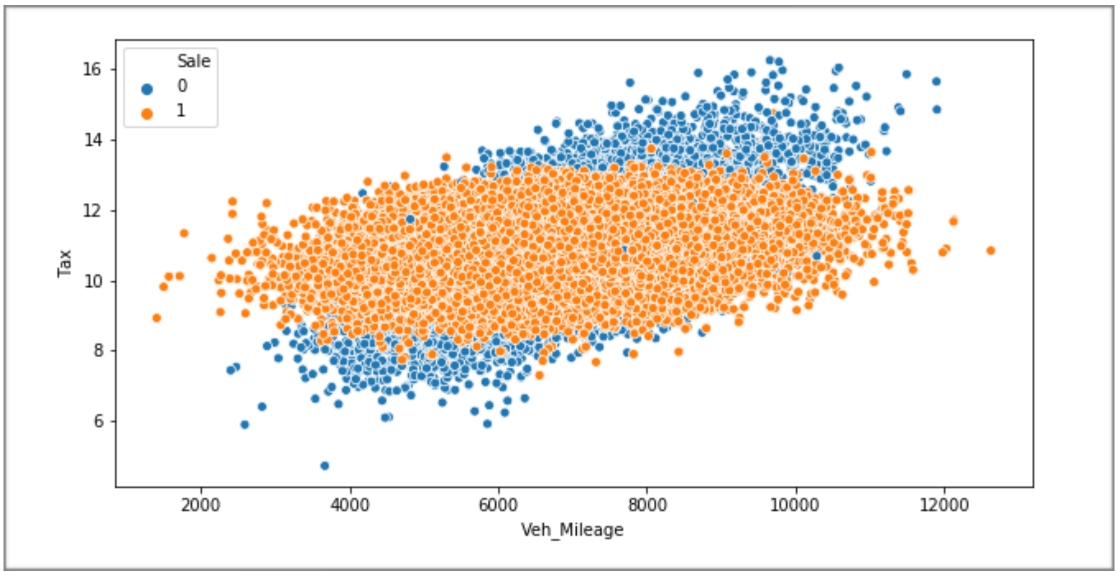
The first and most simple method of imputing missing data was to group the data according to related features and apply means based on the values seen on that group. An example of this would be the imputation of *Veh_Value*. The data was grouped according to the vehicle mileage, price and tax, with these features being binned based on the data's quantiles (0%, 25%, 50%, 75%, 100%) prior to this. After doing so, the mean vehicle value was calculated for each individual group and applied to data with NULL *Veh_Value*.

The choice of which features to group by was made in this case by intuition. In other cases each individual features most correlated counterparts were used to group by.

Imputing via Regression

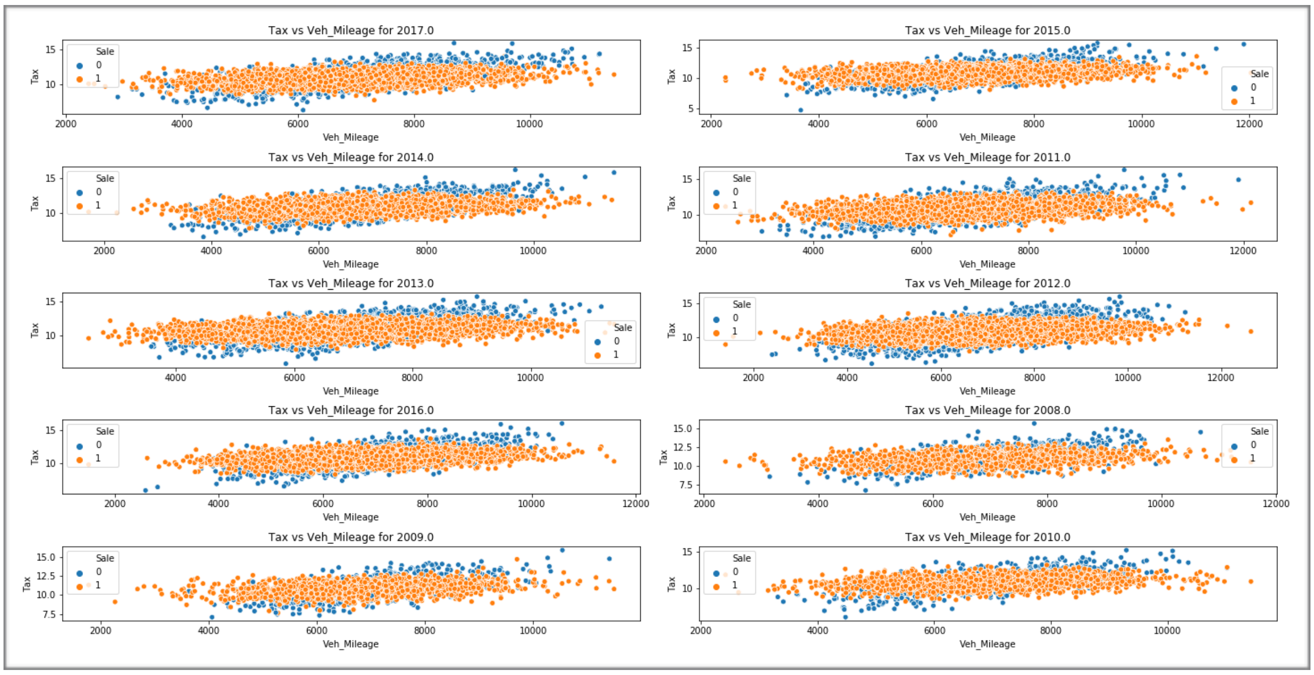
The second attempt at imputing was made by fitting a regression line to two correlated variables and then predicting the missing values based on this regression line. The missing values for Tax were imputed in this manner, with an explanation being supplied below.

Initially the correlations with respect to the Tax feature were checked and it was found that the vehicle mileage was the most highly correlated. These were plotted against each other and can be seen on the following page, split by Sale. An interesting observation was that the two separate groups (sale/no sale) show some form of relationship, with those resulting in no sale showing a quasi-linear increase in tax as the vehicle mileage increases.



Therefore, two separate regression models were fitted to this data with respect to the target so as to predict the missing tax values for entries containing a vehicle mileage feature. This imputation resulted in a slight increase in accuracy and was kept.

Diving further into this, the data was then split according to the year of registration (*Veh_Reg_Year*), showed in the following graph.



Two regression lines were then fit to each individual year of registration, in an attempt to better define the relationship between the two features. This didn't result in an increase in model accuracy, and was therefore not kept instead opting to use the original regression defined previously.

The inverse was also attempted, predicting the vehicle mileage by utilising those rows containing a tax value. Instead of re-fitting another set of regression lines, the same regression line was used as mentioned previously, this time finding x using the line's coefficient and y-intercept value ($y = mx + c$). This also resulted in a decrease in accuracy and was later removed.

Due to the vehicle value feature also showing a positive correlation to the tax feature, the same was done as in the original regression mentioned previously, this time predicting the vehicle value using the tax feature. This showed a slight increase in model accuracy and was therefore kept.

Imputing via Nearest Neighbour

The third and final method used to impute values utilised a nearest neighbour. This was done initially for the highest correlated feature to the target, price. After this it was attempted to be used for other features' missing values but was dismissed as it showed no increase in accuracy.

The features containing price information and all other features were used to fit the nearest neighbour model, with the model in question making predictions for price based on the five nearest neighbours.

Feature Engineering

A method used in previous projects was initially used to create a new feature. The data was grouped based on binned price features (*Price, Veh_Value* and *Tax*), that were binned according to quantile values. Then, for each group, the sum and means of the values that were used to group the data were calculated and added on as new features (having the same names but with *_sum* and *_mean* suffixes). This resulted in a slight drop in accuracy and was therefore removed from the final model.

Another attempt to engineer a feature was made by using a dimensionality reduction algorithm to reduce unimportant features viewed previously. Dimensionality reduction was carried out on *License_Length, Marital_Status, Payment_Type* using a method called multiple factor analysis, which builds PCA/MCA depending on the feature type (in this case it's a mix of continuous and categorial features). The features were reduced to two components and added to the data. This also resulted in a slight drop in accuracy and was therefore removed from the final model.

The final attempt to engineer a feature (and the most simple) included the addition of all of the price related columns as mentioned previously. More specifically, two new features were added which were:

- *Total_Policy_Price*: This feature was the summation of the price and tax features.
- *Total_Value*: This feature was the same as the previous summation but also included the vehicle value.

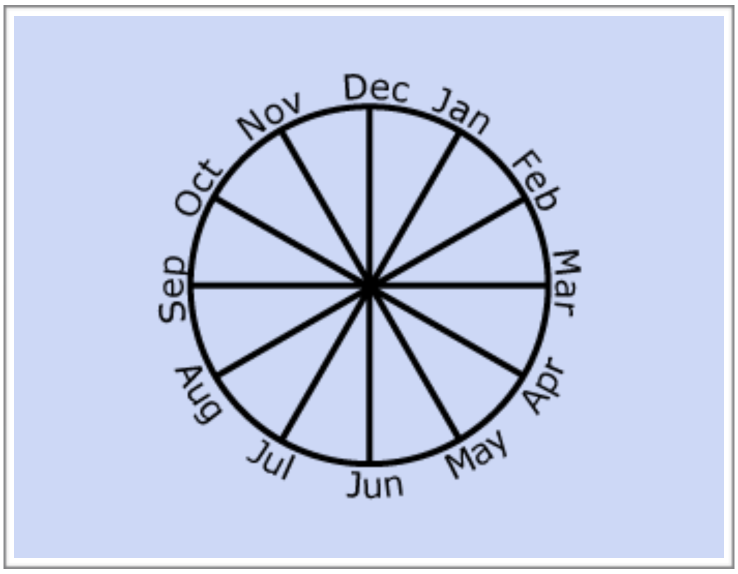
These two features together increased accuracy slightly and were therefore kept in the final model.

Feature Encoding/Model Preparation

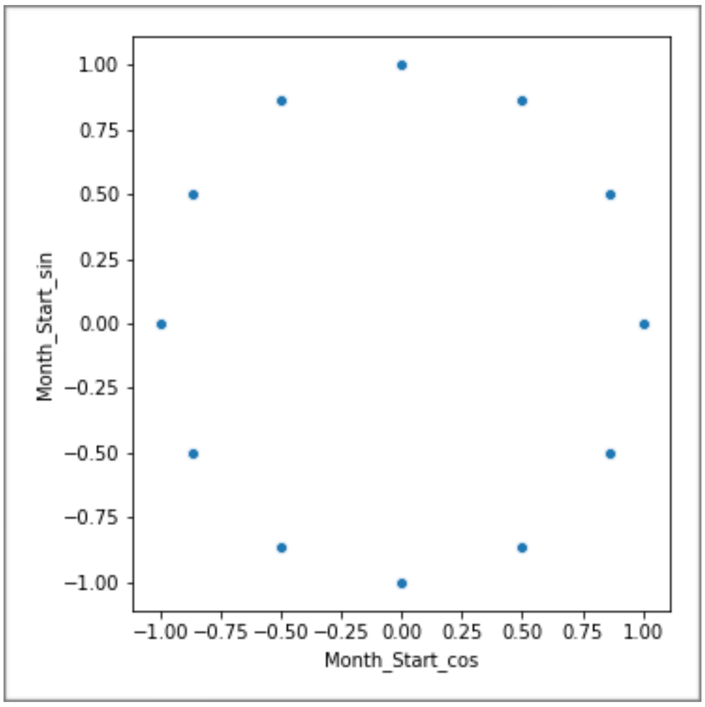
Even though the categorical features (*Marital_Status, Payment_Type*) were removed later on prior to modelling, they were initially used on the first iteration of the created model. These were encoded using *sklearn's* label encoder.

The date information extracted from the original *Date* feature was encoded differently due to its nature. Recall that the date was split into new features *Day_Start, DayOfWeek_Start* and *Month_Start*. Were these features to be encoded using the method mentioned previously, the model would not be able to take advantage of this information fully due to its cyclic nature. Take a label encoded month variable for example; December would be encoded as 12 while January as 1. This creates a 'distance' between the months, something which the model would interpret differently as opposed to understanding them as following one another.

Therefore each cyclic feature was encoded as two separate features (cosine and sine), for them to be represented as a circle and remove that ‘distance’ problem mentioned previously. This can be better understood through the following figures.



With the month features being represented in that manner, seen in the following plot.



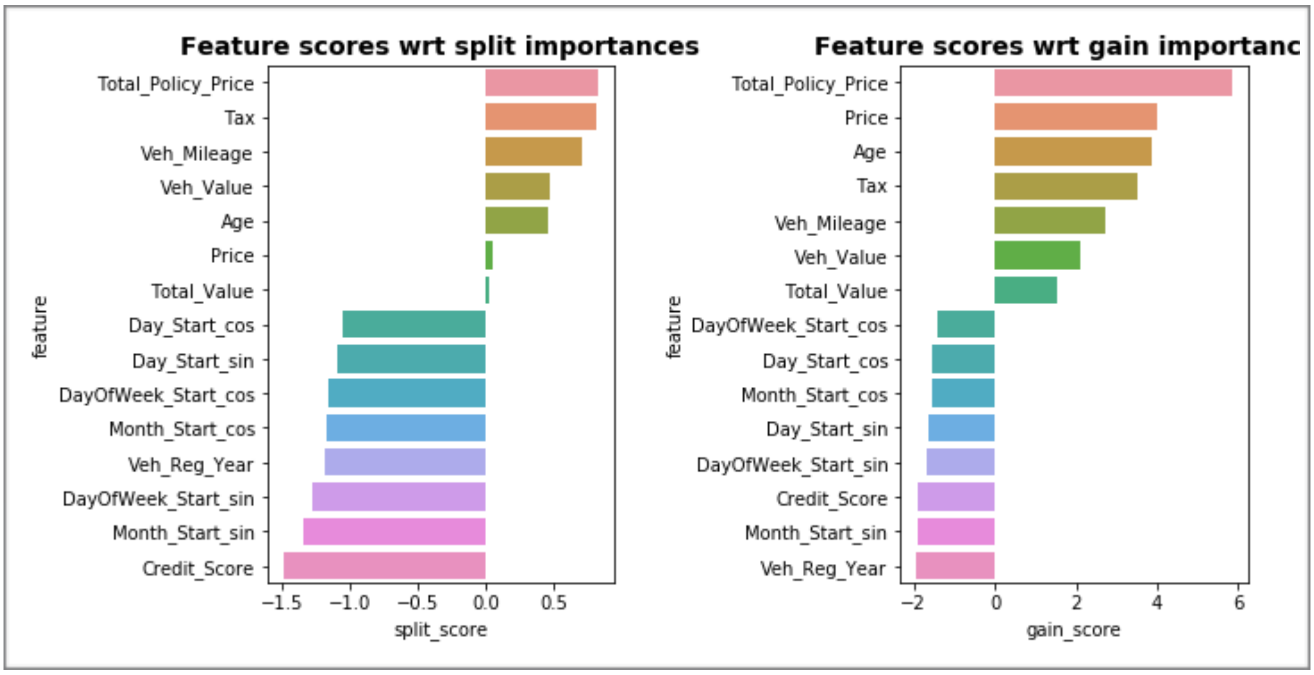
Modelling

As mentioned previously, the model chosen for this classification problem was of the gradient boosting type, more specifically Microsoft’s implementation called LightGBM. Prior to training the model on the data and obtaining metrics to show its accuracy, feature importance was tested using two separate methods.

The first method used was a feature importance metric based on this article, using a lightGBM model in Random Forest setting to fit the data. The steps for this are as follows:

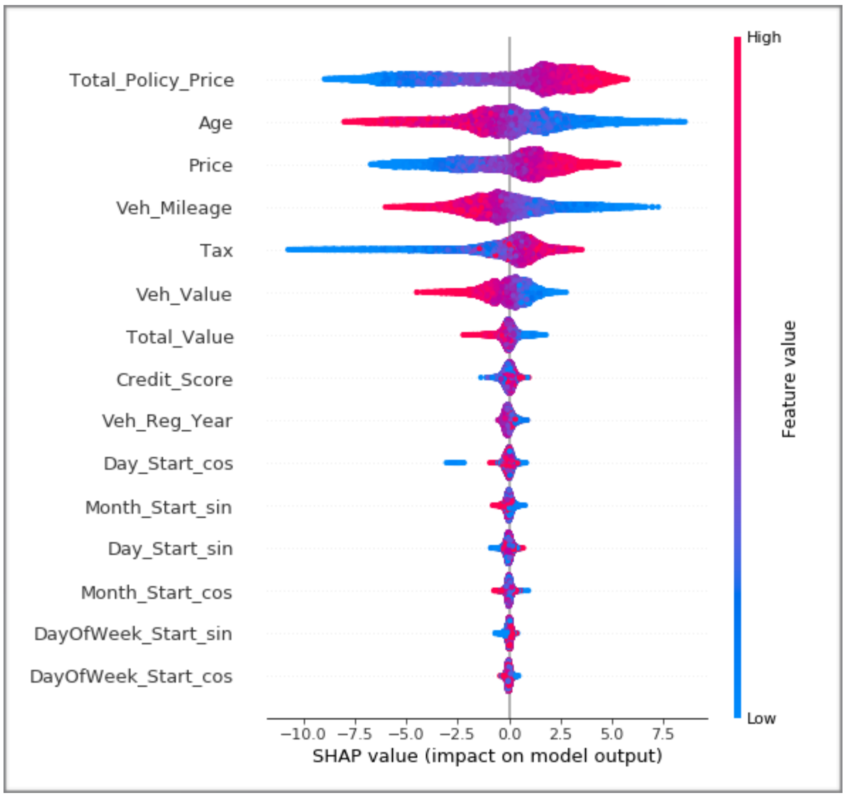
- Create null importance distributions, that is fitting the model on shuffled versions of the target so as to see how the model can make sense of each feature.
- Fit the model again but now on the original data, collecting the feature importances.

- To score each feature compute the ratio of the actual importance to the 75% of the null distributions.



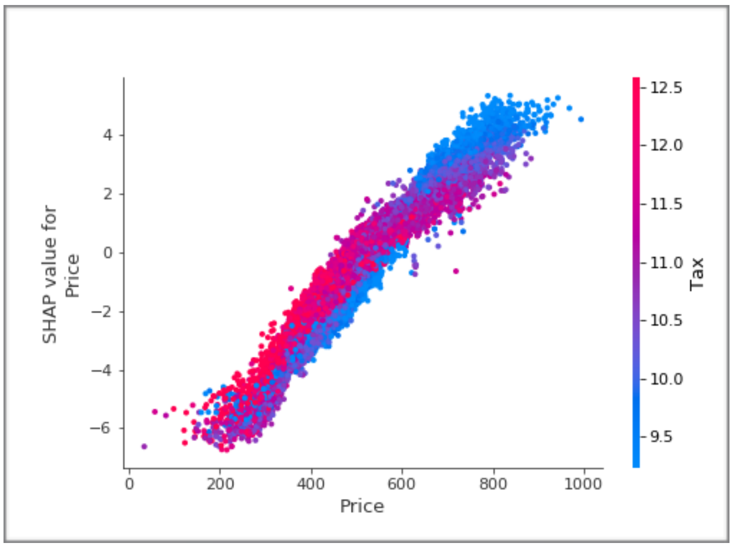
This shows how, individually, the added date features have little to no effect on the model's gain and splitting values.

Another method used to rank features centres around the Shapley value. This method of feature ranking takes a game theory approach, measuring each features contribution to predicting the target variable. For another model such as a linear regression, each feature's importance when making a prediction is the weight of the feature multiplied by the feature value. The shapley ranking method produced the following summary graph ranking features, observed below.



This graph portrays a lot of information. First of all it ranks features based on importance from top to bottom. It also shows the distribution of each feature on the y-axis whilst showing individual data points. Finally it also shows how the distribution affects the predicting power of the model by colouring points that have greater value to the model red.

This can be analysed further by performing a dependance plot on each individual feature. The following plot shows one such plot for the Price feature.



This plot shows the distribution of the price feature with respect to the shap value (importance). Interestingly, the shap value is positive once the price increases beyond 500. Also, as the price increases, the amount of tax is seen to decrease, shown by the points' colour.

HyperParameter Tuning

After initially training the model and obtaining a base accuracy score, the model was run through various iterations each with different internal parameters, so as to obtain the model best suited for this task.

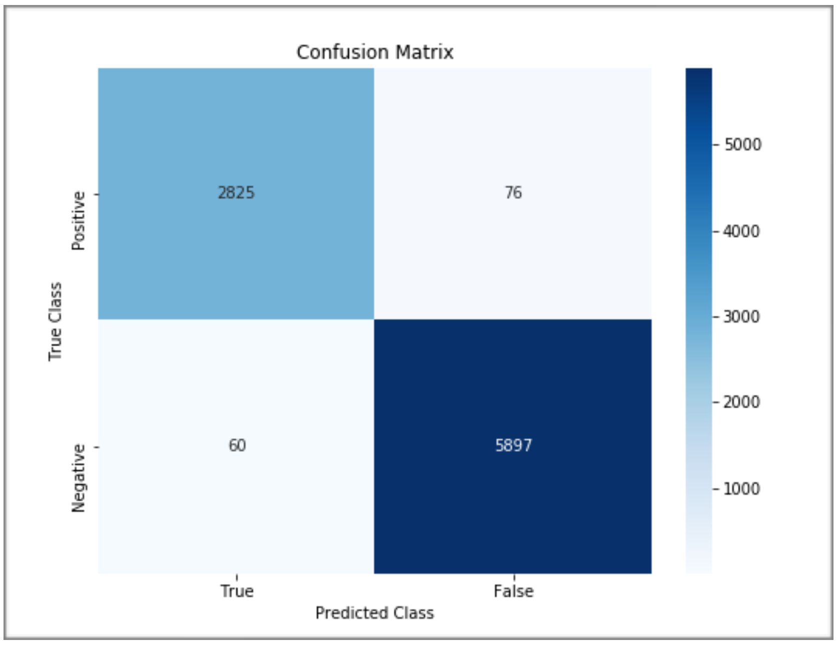
The tuning was done using *sklearn's* RandomizedSearchCV, which iteratively trains the model on the dataset while changing internal parameters based on supplied lists of values. This increased the final accuracy of the model slightly, and as a result the parameters obtained were used in the final version.

Final Accuracy and Result

Prior to training the model the data was split into separate train and test data frames, so as to obtain the accuracy and other metrics on unseen data. These final metrics are shown in the following table.

Accuracy Measure	**Percentage (%)**
Accuracy	98.465
Area Under Receiving Operating Curve (AUC ROC)	99.347
Recall Score	98.993
F1 Score	98.86

Confusion Matrix:

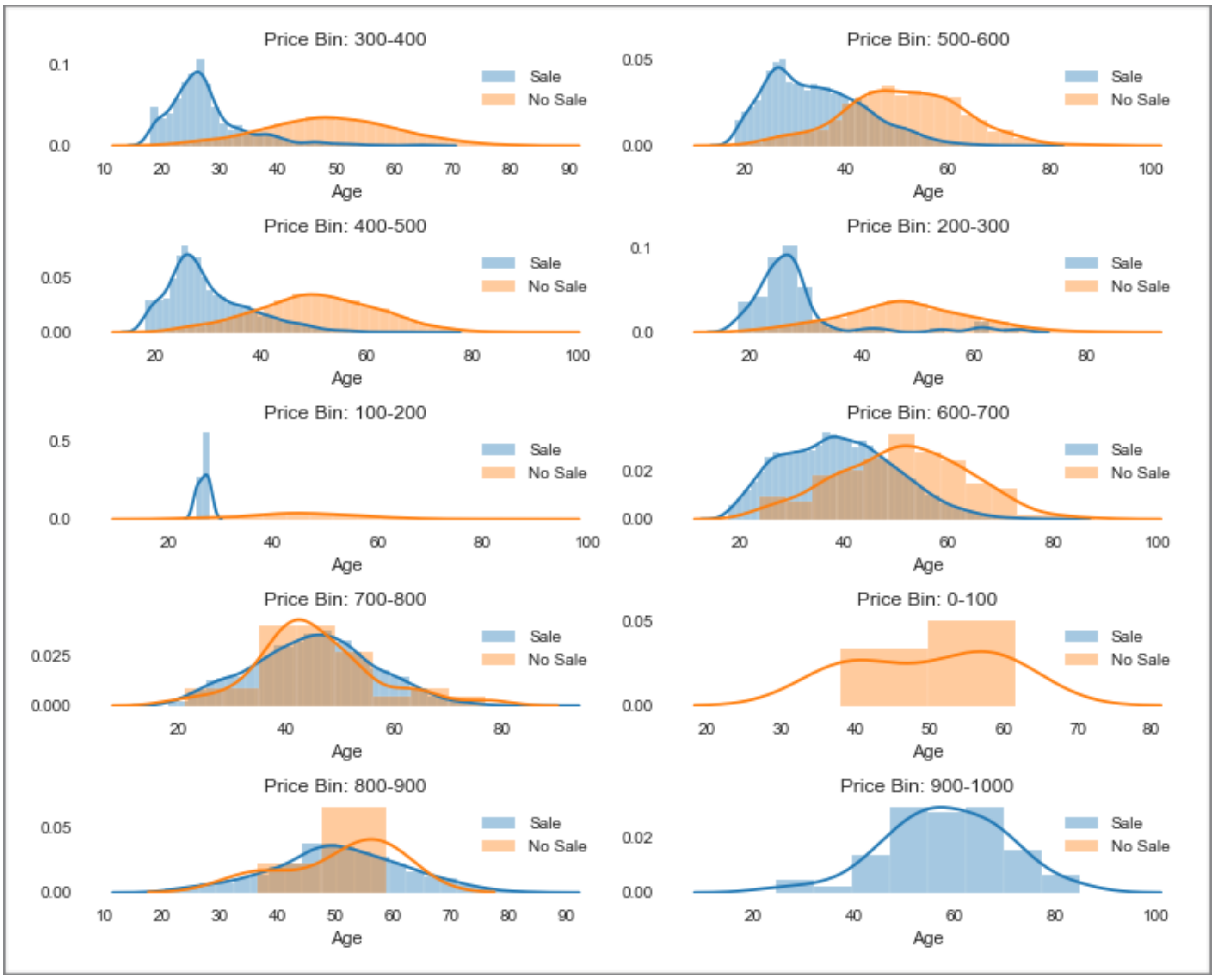


All in all, and when compared to the other simpler models that were trained for comparison (logistic regression produced a base accuracy of ~85%), the results shown here are of a satisfactory nature.

Section 3: Insights

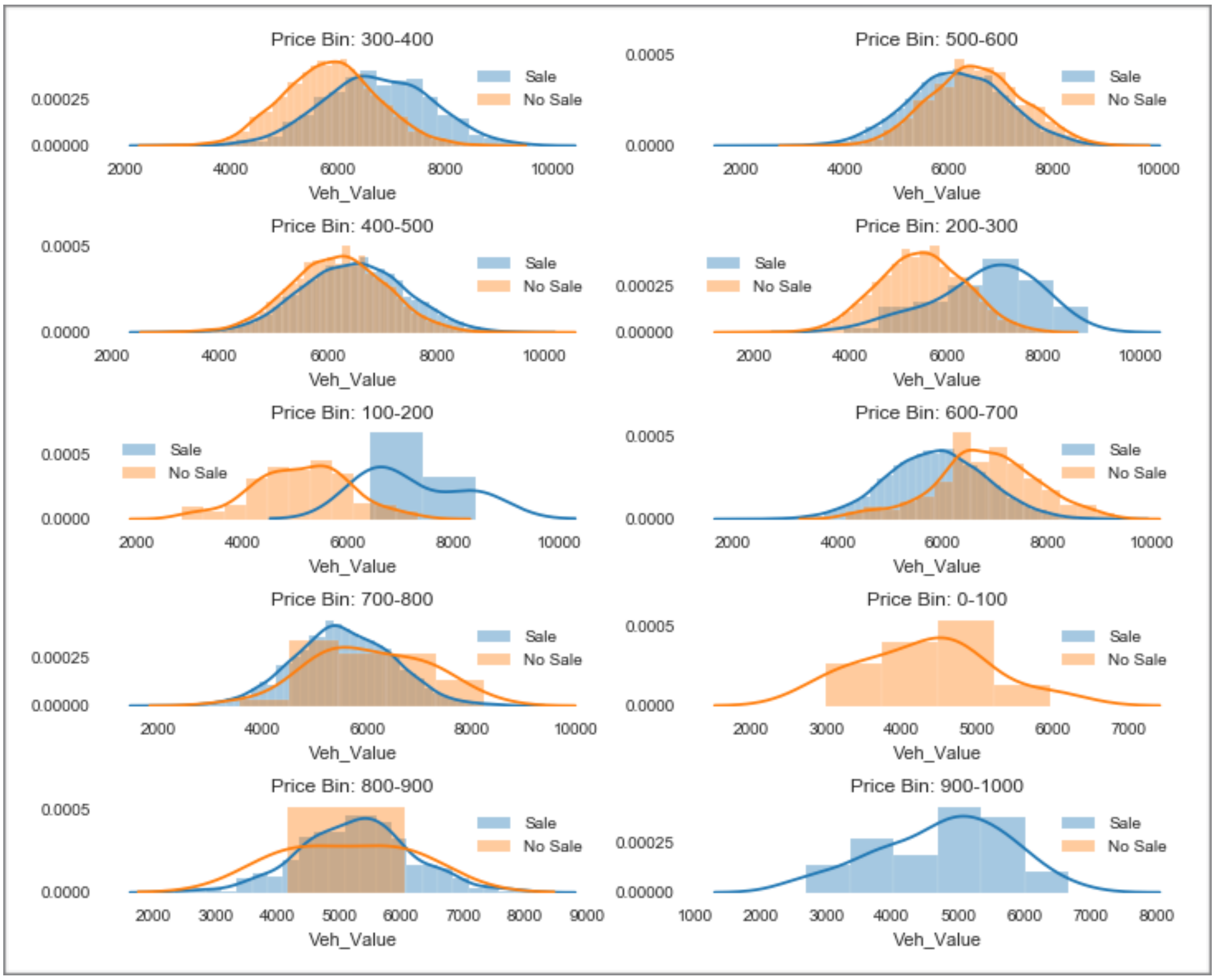
To yield higher sales, one possible method would be to offer some form of promotion to attract new customers. The age feature shows how the distribution of ages is different across the target variable. When looking at the age box-plot shown in the beginning of this document, we can see how the median age of sales to no sales differs. No Sales has a median age of around 50 while Sales are seen to have a median age of around 35. Therefore pushing a new car insurance promotion to the middle aged demographic would be a good starting point.

Let's take a look at the price feature, more importantly the price distribution listed previously. We will take a look at the point when the two distribution lines (sale vs no sale) meet, around the 500 pound range. At this point it is pretty much 50/50 as to if a sale is made or not. Why is that?



It seems as though the lower the price, the more likely that the customer accepting the offer is young. Sales could be improved here by targeting younger customers and pushing more expensive options, or by trying to rope in those older customers that wouldn't normally accept the given policy.

The same plot was created but now with respect to the vehicle value.



Interestingly, there seems to be a relationship between the price and vehicle value. That is, the cheaper the policy price, with a cheaper vehicle it is more likely a sale is not made. The same can be said for more expensive policy prices, expensive vehicles are more likely to result in a no sale.

Although counter-intuitive, a possible approach to gain more income would be to increase the price on (cheap) policies for cheaper vehicles while decreasing those of more expensive vehicles for expensive policies. This could be tested out using the following experimental setup.

Experimental Setup

When dealing with actionable information obtained from data science, it is important to define an experimental setting to test the effect of any defined insights. Therefore a hypothetical situation is defined where a new deal is introduced and the test to be had is if this deal has any affect on the final conversion rate of offered policies.

The main assumption is that all tests are to be done concurrently. Take a random sample of 1000 customers, from which 200 are to be used as a control group. This control group is to be supplied with no offer. From the remaining 800, split them into equal groups defined by the deal in question together with any other deals that have already been put in place. A non-parametric test such as a T-test is to be carried out on each individual group to see if statistically the new group has had an affect on the conversion rate.

Summary

In summary, the following steps were carried out to complete this assessment.

1. Initially, some analysis was performed on the data to better understand the problem at hand.
2. Missing data was imputed where it added to the accuracy of the base model. This was done using a variety of different methods.
    - First, the dataset was grouped according to features that intuitively should have shown some form of relationship to the missing feature. Then mean/median values were applied to the missing feature based on the groups.
    - The second method tried involved fitting regression lines between two correlated variables, then using that regression model to fill in the missing data.
    - The last method involved implemented a nearest neighbour approach to fill in missing data, based on the five closest neighbours.
3. Some feature engineering was carried out. This was done by grouping data according to the monetary features (*Veh_Value, Tax, Price*) and applying the mean and sum of each to each group. When this was shown to not increase the model's performance, dimensionality reduction was carried out on a subset of features that also did not add to the model's accuracy. This was done with the hope that exposing the principal component of these features might bring some new information for the model to act upon. This was also shown to not add to the model's performance. Finally, two new features were added that constituted the addition of *Price* and *Tax* and *Price, Tax* and *Veh_Value*. These new features contributed to a slight increase in accuracy and were therefore kept for the final model.
4. Encoding of the required features was then carried out, using a cyclic encoding method for the date information that represented them as two components (*sin* and *cos*) for the model to take advantage of their cyclic nature.
5. Feature ranking was done via two methods.
    - The first included training the model while supplying dummy target variables, and then inspecting the *gain* and *split* criteria for each feature. This was done so as to see what the model makes of the data without being told the target. Then the model was trained on the proper data, with the gain/split scores being compared to the dummy training.
    - The second method to perform feature ranking involved using a technique revolving around the SHAP value. That is, measuring each feature's individual contribution into making a prediction.
6. Hyperparameter tuning was done via *sklearn's* RandomGridCV, which randomly iterates over pre-defined hyper parameter values whilst training the model on the data each time. The parameters which led to the best accuracy were then kept.
7. The final accuracy of the model was seen to be 98.465%, with all other metrics used to grade the model showing satisfactory results.
8. Some insights about increasing sales and income were shown, with an experimental setup being described that could be used to test any possible new promotions and their effects on sales.