**Advances Data Science/Architecture: ‘D’ Predictors**

**Final Project Report**



**Authors**

Apoorva Verma, Bhushan Patil, Gaurav Saxena, Shruti Ranjalkar

Date: 27 April 2016

Version 1.0

Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 04/11/2016 | 0.1 | Draft Version | Apoorva Verma, Bhushan Patil, Gaurav Saxena, Shruti Ranjalkar |
| 04/14/2016 | 0.2 | Added/ Amended topics: Feature Selection | Apoorva Verma, Bhushan Patil, Gaurav Saxena, Shruti Ranjalkar |
| 04/18/2016 |  | Added/ Amended topics: Classification Model 1, Feature Selection | Apoorva Verma, Bhushan Patil, Gaurav Saxena, Shruti Ranjalkar |
| 04/22/2016 | 0.3 | Added/ Amended topics: Classification Model 2 | Apoorva Verma, Bhushan Patil, Gaurav Saxena, Shruti Ranjalkar |
| 04/26/2016 | 1.0 | Added/ Amended topics: Regression Technique | Apoorva Verma, Bhushan Patil, Gaurav Saxena, Shruti Ranjalkar |

Table of Contents

1. **Application of Data Science in Insurance**
2. **Problem Statement**
3. **Implementation**
4. **Data Preparation**
5. **Session History**
6. **Feature Engineering**
7. **Python Code for Data Preprocessing**
8. **Data Split**
9. **Feature Selection for Policy prediction**
10. **Feature Importance**
11. **Model-1: Classification**
12. **Model-2: Classification**
13. **Model-3: Linear Regression**
14. **Conclusion**

**Application of Data Science in Insurance:**

By integrating data science into customer retention strategies, insurers can provide the early warning that agents, call center representatives and other employees need to keep their best customers longer and improve customer lifetime value. Predictive analytics provides the ability to:

• Discover policy termination patterns and profiles of customer who leave for a deeper understanding of why they left

• Predict future customer value to determine if you want to retain an at-risk customer, and at what cost

• Predict what offer or service would prevent a customer from switching insurers, and what price will cover your risk of insuring that customer for another term Likewise, integrating predictive analytics into customer profitability strategies uncovers complex purchasing behaviors, identifies events that predict policyholder needs, and increases marketing effectiveness by isolating the best targets and best offers for customer microsegments. These analytic insights also help insurers capitalize on opportunities at each customer interaction point whether it’s through agents, call centers, by email or online. Predictive analytics gives insurers the tools to:

• Alert agents and other customer facing employees to opportunities as they occur – by integrating predictive insights into customer interaction systems and processes

• Monitor customer behavior for events that indicate potential needs – and provide the right cross-sell or up-sell offer in real time

• Automatically choose the best delivery channel for each offer and customer to increase acceptance

**Problem Statement:**

Predict a purchased policy based on transaction history

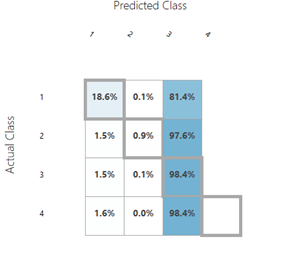
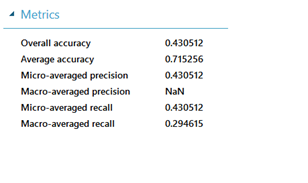
Predict the purchased coverage options using a limited subset of the total interaction history. If the eventual purchase can be predicted sooner in the shopping window, the quoting process is shortened and the issuer is less likely to lose the customer's business.

Using a customer’s shopping history, predict what policy they will end up choosing and the price for that policy.

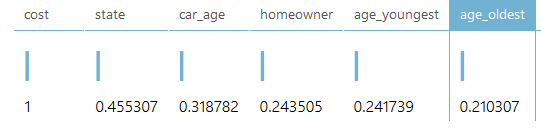
**Implementation-**

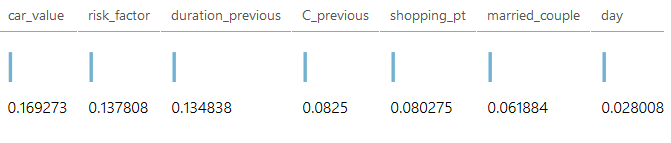
**Data Preparation**

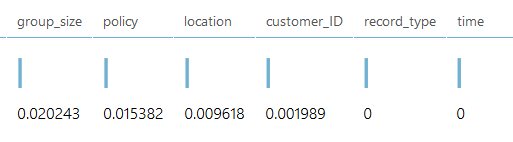
Initially, we had built a model using Multiclass Decision Jungle Classifier using Microsoft ML Azure tool without any data preparation and feature selection. The results are:



We observed the correlation between the features:







Derived few new features:

Age\_Diff, Mean\_Age, Old\_Person, FamilyPlan, newCar, oldCar.

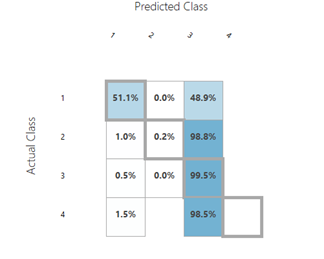
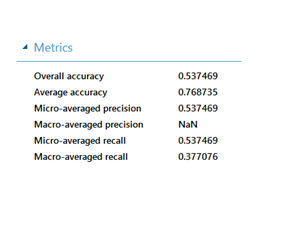
After performing this Data preprocessing, we have used Multiclass Decision Forest which increased the overall accuracy.

We have replaced the N/A values to zero

There are 3 policies of handling NA values:

1) Totally removing the rows, this policy can reduce population features thus that not advisable. 2) Replacing the values with the mean, this scheme also has some drawbacks. Here as the mean is already concentrated and it will show some unnecessary pattern in data thus it may also harm the population.

3) Putting the values 0, this also shows some random pattern but we went with this strategy as this is less harmful as other two.



**Result:** The classes are imbalance and without proper data preparation and features from the session history the model is not able to predict with precision. The features derived also seems unimportant. Because the model predicts class 3 always.

After Evaluating basic components and previous models, we have selected the next model

**Session History:**

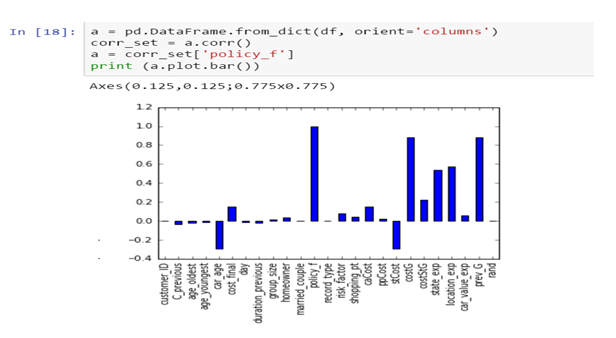
The importance of the session history from business perspective suggests following:

* Most Customers bought the policy that they have visited in last session.
* Every State has a particular policy purchasing pattern.
* According to business, the Car\_Age determines the amount a customer spends on Insurance policy.

**Feature Engineering:**

1. **New Features from Session History and Training Data (For Policy Prediction):**

* Previous Visited Policy
* Mean of Cost per policy and State( CostStG:- here we found the mean of cost per policy first, mean of cost per state. Then for every customer we took a mean of both these means. )
* Pattern of Policy per State (Mean of Policy per State).
* Pattern of Policy per location (Mean of Policy per Location).
* Pattern of Policy per Car\_Value (Mean of Policy per Car\_Value).

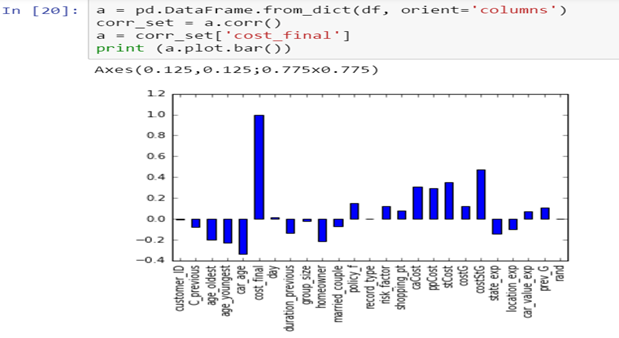
****

**-** High Co-relation found between cost per policy and previous policy.

- The feature we removed here is the cost per policy because previous policy describes the session attributes.

1. **New Features from Session History and Training Data (For Cost Prediction):**

* Previous visited cost and policy.
* The Policy predicted by the Final Model.
* For every Policy Mean Value of Cost per State. ( CostStG:- here we found the mean of cost per policy first, mean of cost per state. Then for every customer we took a mean of both these means. )
* Mean Value of Cost per State.
* Mean Value of Cost per Policy.
* Mean Value of Cost per car age
* For every Customer Id Mean Value of Cost per person (transformed using the group\_size)



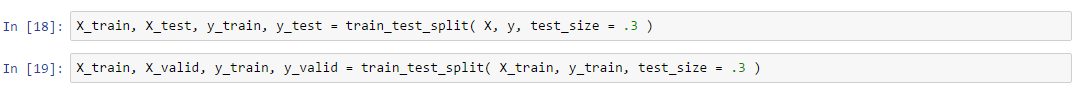
**Python Code for Data Preprocessing**



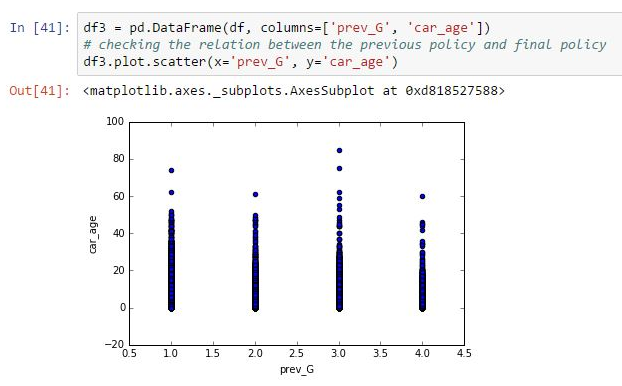
 

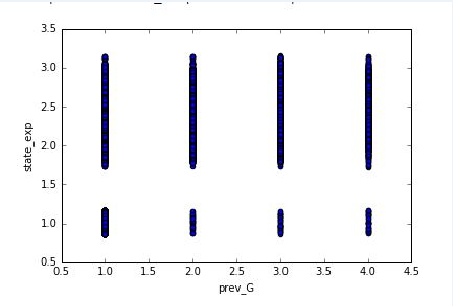
**Data Split:**

We split the data into test train and validation set. 30% of training data is test data set and 30% is validation set.



**Feature Selection for Policy prediction:**

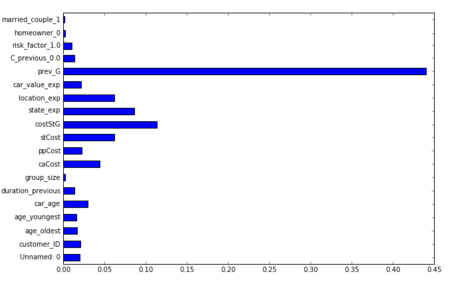




Features Selected for Policy Prediction:

Married\_Couple, Home\_Owner, Risk\_Factor, C\_Previous, Previous\_Policy (prev\_G), Car\_Value\_Exp, Location\_Exp, State\_Exp, Car\_Age

**Feature Importance:**

****

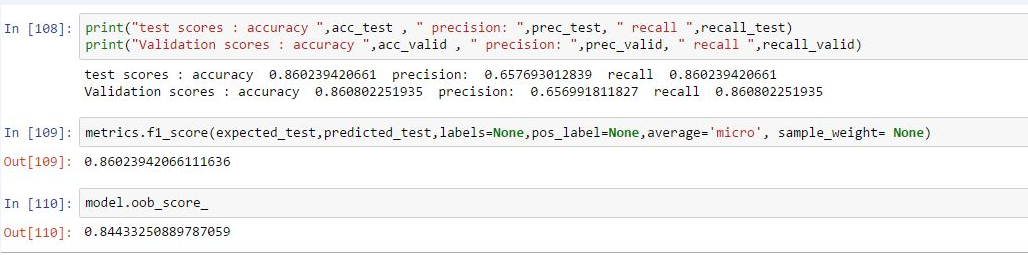
**Model-1: Classification**

**Random Forest Multiclass-Classifier:** Random forests is a notion of the general technique of random decision forest that are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for classification, [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks, that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean prediction (regression) of the individual trees. It uses multiple learning algorithms to obtain better [predictive performance](https://en.wikipedia.org/wiki/Predictive_inference) than could be obtained from any of the constituent learning algorithms.

**Bagging:** It is a [machine learning ensemble](https://en.wikipedia.org/wiki/Ensemble_learning) [meta-algorithm](https://en.wikipedia.org/wiki/Meta-algorithm) designed to improve the stability and accuracy of [machine learning](https://en.wikipedia.org/wiki/Machine_learning) algorithms used in [statistical classification](https://en.wikipedia.org/wiki/Statistical_classification) which also improves the modeling by reducing the variance and overfitting chances.

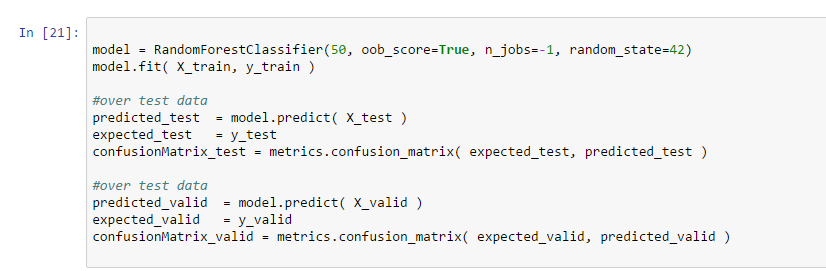
The above algorithm is provided in python and by applying this method we get a predicted value from multitude of the decision trees and for selecting the policy the method uses bagging. It takes out the probability from all the trees and assigns the policy to the feature corresponding to which many trees have voted. Thus reduces the chances of overfitting.

**Model Fitting:**

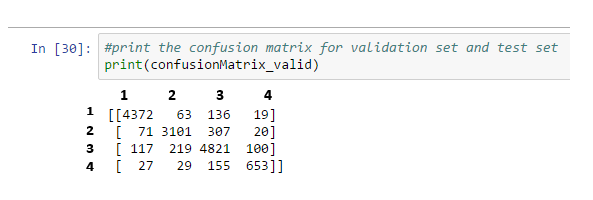


After Tuning the Parameters of Random Forest: Number of Trees: 50, N\_Jobs=-1 and random\_state=42 we achieved test scores:

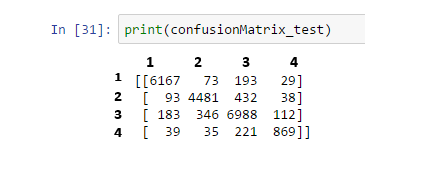
**Accuracy 91%, Precision: 89%, recall: 91% and F1-Score: 91%.**

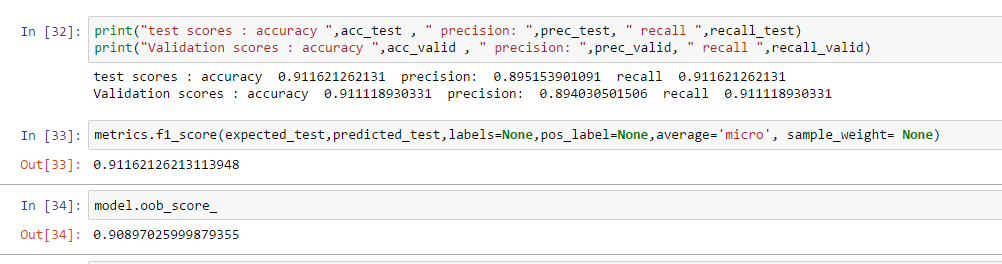


**On the Validation set after feature selection:**



**On the test separated data:**





**Model 2:**

**Implemented Algorithms:**

1. **‘Multiclass Logistic Regression’ Algorithm:** Logistic regression is a well-known method in statistics that is used to predict the probability of an outcome, and is particularly popular for classification tasks. The algorithm predicts the probability of occurrence of an event by fitting data to a logistic function. In multiclass logistic regression, the classifier can be used to predict multiple outcomes.

1. **‘Multiclass Decision Forest’ Algorithm:** The algorithm works by building multiple decision trees and then *voting* on the most popular output class. Voting is a form of aggregation, in which each tree in a classification decision forest outputs a non-normalized frequency histogram of labels. The aggregation process sums these histograms and normalizes the result to get the “probabilities” for each label. The trees that have high prediction confidence will have a greater weight in the final decision of the ensemble.

**Implementation of model:**

* Data split into 70:30 ratio

70: Train Model

30: Score Model

Splitting mode: Split Rows

Fraction of rows in the first output of dataset: 0.7

Randomized Split: Yes

* **Multiclass Logistic Regression:**

Parameters:

Create Trainer Mode: Single Parameter

Optimization Tolerance: 1E-07

L1 Regularization Weight: 2

L2 Regularization Weight: 2

Memory size for L- BFGS: 20

Allow unknown Categorical Values: Yes

**Multiclass Decision Forest:**

Resampling Method: Replicate

Create trainer mode: Single Parameter

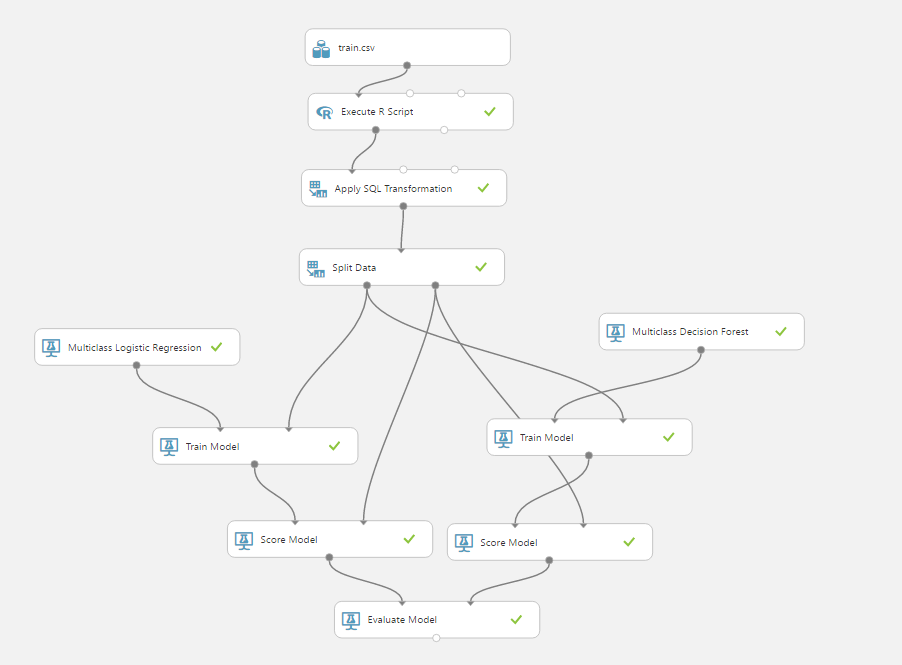
No. of Decision trees: 10

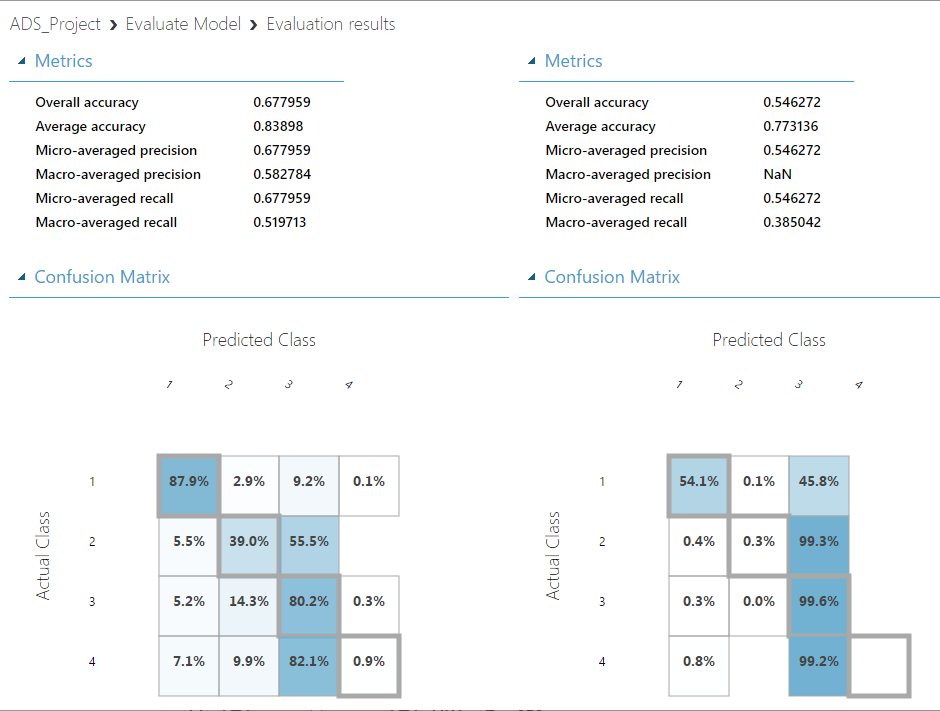
Maximum depth of the decision trees: 32

No. of random splits as per node: 90

Minimum number of samples per leaf node: 1

**Evaluation of Model:**



**Performance of the model-**

Multiclass Logistic Regression model is suitable for the business problem.

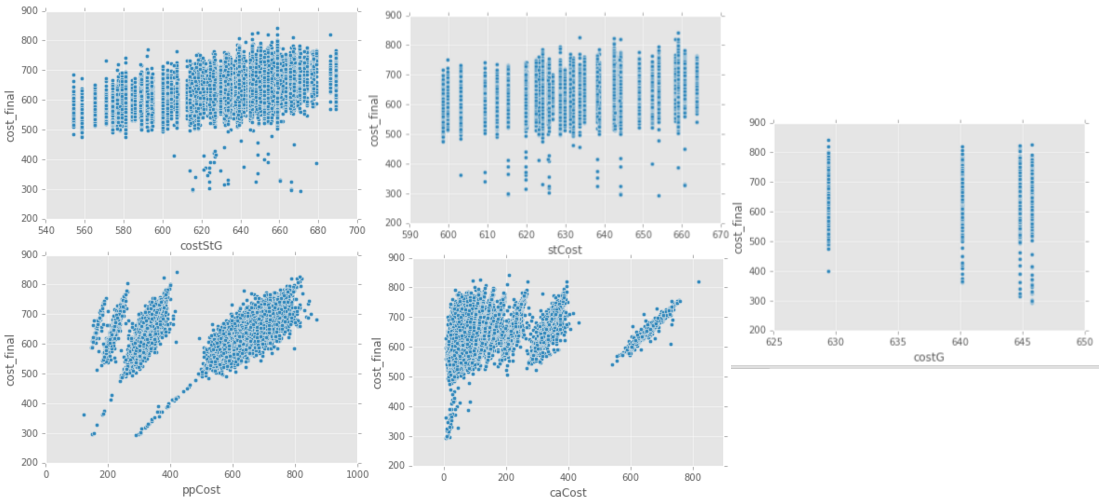
**Reasons:**

1. High accuracy as compared to Multiclass Decision Forest Algorithm (Overall Accuracy: 83.89%)

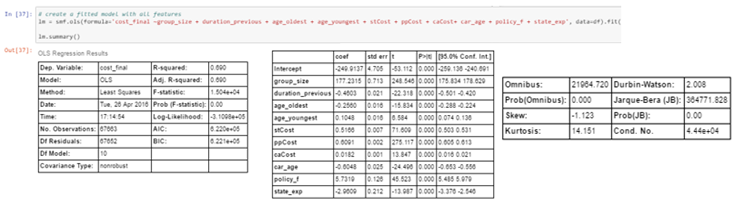
2. Detection of 4th policy in the Multiclass logistic regression model

**Model-3: Linear Regression**

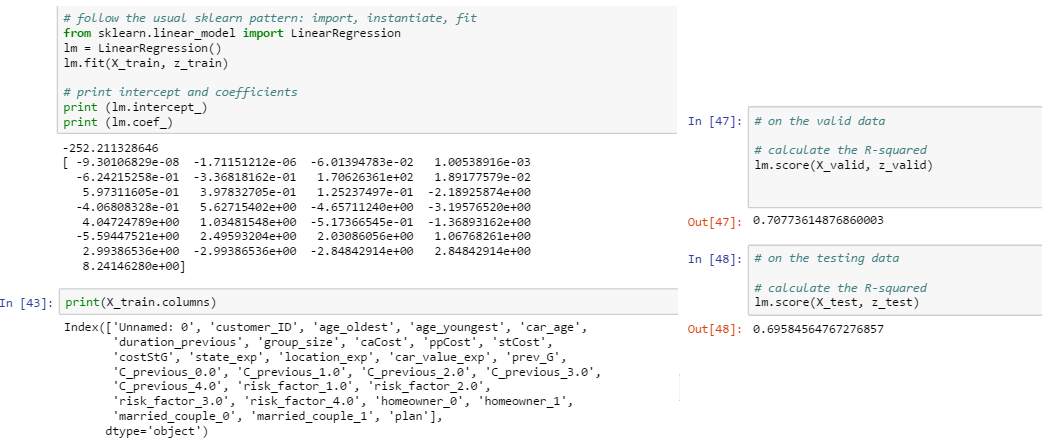
**Feature Selection:**

****

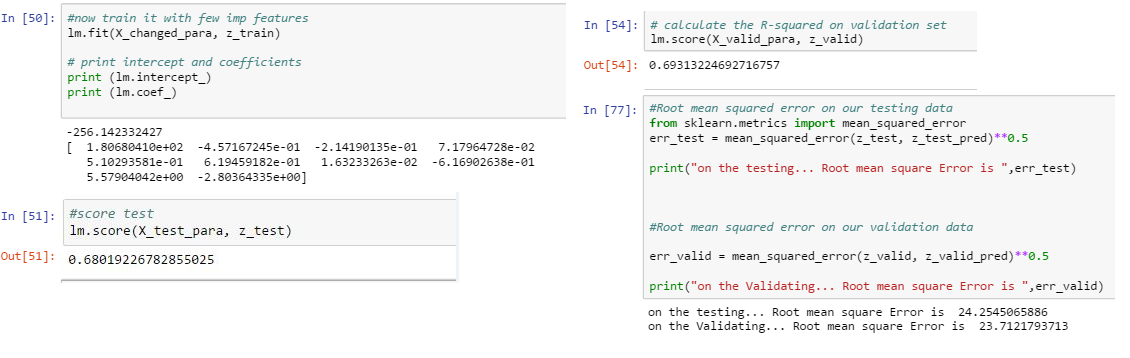
* Applied linear regression of Statsmodels recursively to get least AIC and BIC



Without Feature Selection, we got the following regresssion model which gave R-Squarred of 0.70 on validation set and 0.69 on testing Set. But AIC and BIC in that case was 6.536



-Implemented feature selection as interpreted by AIC and BIC of Statsmodels and build the final model. The Akaike information criterion (AIC) and the Bayesian information criterion (BIC) are based on the log-likelihood of number of parameters. Both measures introduce a penalty for model complexity, but the AIC penalizes complexity less severely than the BIC.



We achieved a linear regression model which gave a **root mean square error of 24**.

**Modeling Conclusion:**

**Final Model Selected:**

Random Forest Multiclass Classifier performed the best out of the classification models and provided:

**Accuracy 91%, Precision: 89%, recall: 91% and F1-Score: 91%.**

For Regression/policy price prediction we choose the Linear Regression of SkLearn , which provided a **Root mean squared Error of 24.**

**Conclusion:**

We found out that the Cost per policy per state for every customer ID and their previous visited policy played out to be most important.

* Accuracy: 71.9%
* Recall: 71.9%
* F- Measure: 71.9%
* Root Mean Square Error: 28.89
* We recalled 502 instances of Class 4.

And the confusion matrix:

