**Retail Sales Prediction**

**Lovejeet Singh**

**Data science trainee,**

**AlmaBetter, Bangalore**

**Abstract:**

Rossmann operates over 3,000 drug stores in 7 European countries. This paper outlines a retail sales prediction. The relative importance of consumer demographic characteristics for accurately modeling the sales of each customer type are derived and implemented in the model. Data consisted of daily sales information at the store type

Our experiment can help understand what could be the feature that can affects our different store type sales by feature selection, feature engineering, data analysis and prediction with machine learning algorithms taking into account previous trends to determine the correct model.

1. **Problem Statement**

Data provided by Rossmann Stores available which is operate in whole Europe. The product range includes up to 21,700 items and can vary depending on the size of the shop and the location. In addition to drugstore goods with a focus on skin, hair, body, baby and health, Rossmann also offers promotional items ("World of Ideas"), pet food, a photo service and a wide range of natural foods and wines.

The main objective is to build a predictive model, which could help them in predicting the Sales. This would in turn help them to estimating the product demand and supply according to customers’ needs on different store type located at different locations.

**The Rossman company provides us two datasets. Each dataset holds different features.**

* Rossmann Stores Data.csv - historical data including Sales
* store.csv - supplemental information about the stores

**Following are the features insides in our two-dataset provided by Rossman:**

* + Id - an Id that represents a (Store, Date) duple within the test set.
  + Store - a unique Id for each store
  + Sales - the turnover for any given day (this is what you are predicting)
  + Customers - the number of customers on a given day.
  + Open - an indicator for whether the store was open: 0 = closed, 1 = open.
  + State Holiday - indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None.
  + School Holiday - indicates if the (Store, Date) was affected by the closure of public schools
  + Store Type - differentiates between 4 different store models: a, b, c, d.
  + Assortment - describes an assortment level: a = basic, b = extra, c = extended
  + CompetitionOpenSince [Month/Year] - gives the approximate year and month of the time the nearest competitor was opened
  + Promo - indicates whether a store is running a promo on that day.
  + Promo2 - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating

**Following are the library we will use in our analysis and model building: -**

* Pandas: - Pandas is for solving data wrangling and exploratory.
* NumPy: - NumPy is for numerical problem solving
* Matplotlib: - Matplotlib is for Data Visualization
* Seaborn: - Seaborn is for Data Visualization
* Scikit Learn: - Scikit learn for Machine learning

1. **Steps involved:**

* **Exploratory Data Analysis**

After loading the datasets, we performed this method by comparing our target variable that is Sales with other independent variables. This process helped us figuring out various aspects, distribution and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

* **Null values Treatment**

Our Store.csv dataset contains a large number of null values which might tend to disturb performance of our model. hence, we dropped those features which has more than 40% null values to get a better result and the feature which has less than 40% null values we replace them using the mean, mode and median of these features according to their distribution.

* **Encoding of categorical columns**

We used One Hot Encoding to produce binary integers of 0 and 1 to encode our categorical features because categorical features that are in string format cannot be understood by the machine and machine learning algorithm and needs to be converted to numerical format.

* **Feature Selection**

In these steps we used Liner Regression that’s why we use Variance Inflation Factor (VIF) to check the multicollinearity of each feature. And also check the correlation of each independent variable with target variable i.e., which feature is more important compared to our model and which is of less importance.

Next, we use all the dataset along with dummies encode categorical variable for our other regression algorithm like Decision tree random forest etc. these algorithms automatically handle multicollinearity and correlation.

* **Standardization of features**

Our main motive through this step was to scale our data into a uniform format that would allow us to utilize the data in a better way while performing fitting and applying different algorithms to it. To scale our data, we use StandardScaler which is available in preprocessing class present in scikit learn library.

* **Fitting different models**

For modelling we tried various regression algorithms like:

* 1. Liner Regression
  2. Liner Regression with Regularization lass, ridge and elastic net.
  3. Decision Tree
  4. Random Forest
  5. XGBoost
  6. Gradient Boosting
* **Tuning the hyperparameters for better accuracy**

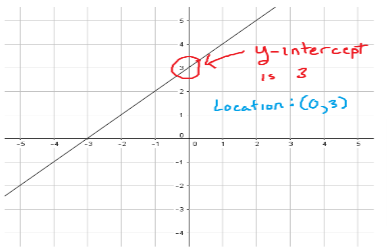
Tuning the hyperparameters of respective algorithms is necessary for getting better performance and to avoid overfitting in case of tree-based models like Random Forest Regressor, XGBoost Regressor and Gradient Boosting.

1. **Algorithms:**
2. **Liner Regression**

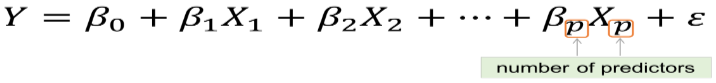
Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.



where Y: output or target variable  
X: input/dependent variable  
β1: Intercept  
β2: constant of X



**Multiple Linear Regression**: it’s simple as its name, to elucidate the connection between the target variable and two or more explanatory variables. Multiple linear regression is used to do any kind of predictive analysis as there is more than one explanatory variable.



**KEY ASSUMPTIONS IN REGRESSION MODEL:**

• The dependent/target variable is continuous

• There isn’t any relationship between the explanatory/independent variables (no multicollinearity)

• There should be a linear relationship between target/dependent and explanatory variables

• Residuals should follow a normal distribution

• Residuals should have constant variance

• Residuals should be independently distributed/no autocorrelation

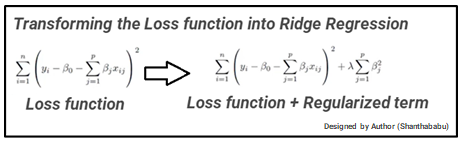
1. **Liner Regression with Regularization lass, ridge and elastic net**

During the Machine Learning model building, the Regularization Techniques is an unavoidable and important step to improve the model prediction and reduce errors.

**1.** Ridge Regression (L2 Regularization):

Basically here, we’re going to minimize the sum of squared errors and sum of the squared coefficients (β). In the background, the coefficients (β) with a large magnitude will generate the graph peak and deep slope, to suppress this we’re using the lambda (λ) use to be called a  
Penalty Factor and help us to get a smooth surface instead of an irregular-graph. Ridge Regression is used to push the coefficients(β) value nearing **zero**in terms of magnitude. This is L2 regularization, since its adding a penalty-equivalent to the **Square-of-the Magnitude** of coefficients.

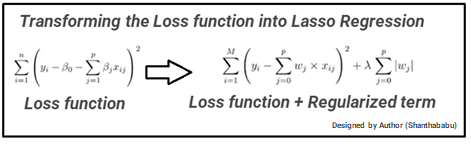
*Ridge Regression = Loss function + Regularized term*

.

**2**. Lasso Regression (L1 Regularization):

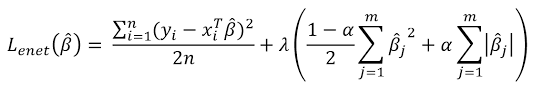
This is very similar to Ridge Regression, with little difference in Penalty Factor that coefficient is magnitude instead of squared. In which there are possibilities of many coefficients becoming zero, so that corresponding attribute/features become zero and dropped from the list, this ultimately reduces the dimensions and supports for dimensionality reduction. So, which deciding that those attributes/features are not suitable as predators for predicting target value. This is L1 regularization, because of adding the **Absolute-Value** as **penalty-equivalent**to the magnitude of coefficients.

***Lasso Regression = Loss function + Regularized term***



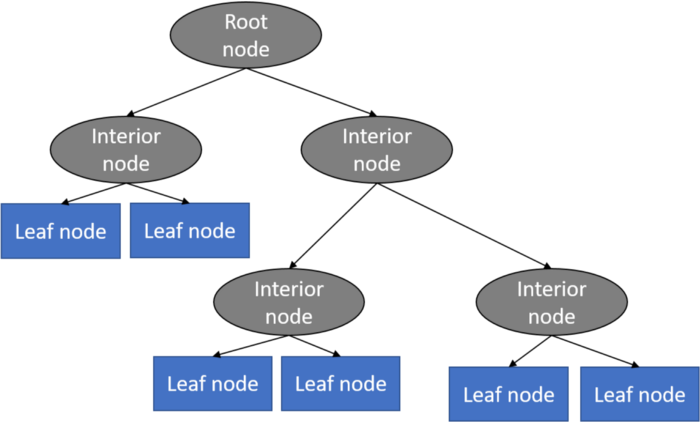
**3.** Elastic-Net Regression Regularization:

Elastic net is a penalized linear regression model that includes both the L1 and L2 penalties during training. Using the terminology from “The Elements of Statistical Learning,” a hyperparameter “alpha” is provided to assign how much weight is given to each of the L1 and L2 penalties.



1. **Decision Tree**

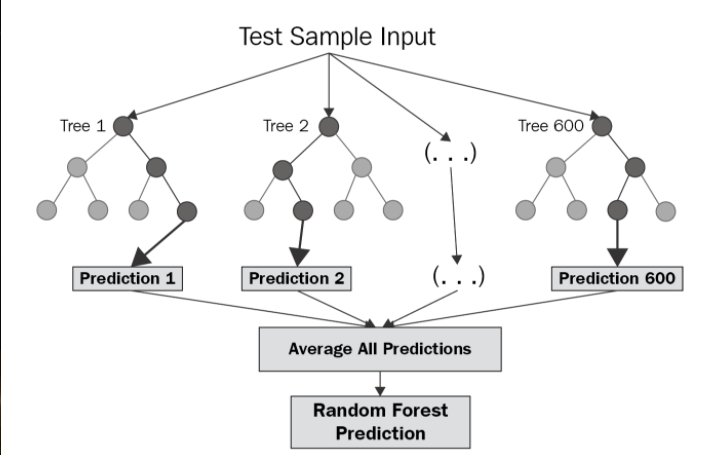
Decision Tree is one of the most commonly used, practical approaches for supervised learning. It can be used to solve both Regression and Classification tasks with the latter being put more into practical application.



It is a tree-structured classifier with three types of nodes. The Root Node is the initial node which represents the entire sample and may get split further into further nodes. The Interior Nodes represent the features of a data set and the branches represent the decision rules. Finally, the Leaf Nodes represent the outcome. This algorithm is very useful for solving decision-related problems

1. **Random Forest Regression**

Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.



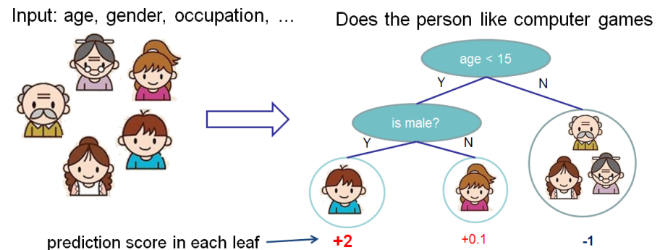
The diagram above shows the structure of a Random Forest. You can notice that the trees run in parallel with no interaction amongst them. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees Pick at random *k* data points from the training set. Build a decision tree associated to these *k*data points. Choose the number *N*of trees you want to build and repeat steps 1 and 2. For a new data point, make each one of your *N*-tree trees predict the value of *y* for the data point in question and assign the new data point to the average across all of the predicted *y*values.

1. **XGBoost**

XGBoost is a powerful approach for building supervised regression models. The validity of this statement can be inferred by knowing about its (XGBoost) objective function and base learners. The objective function contains loss function and a regularization term. It tells about the difference between actual values and predicted values, i.e how far the model results are from the real values. The most common loss functions in XGBoost for regression problems is reg:linear.

1. **Gradient Boosting**

Gradient boosted trees consider the special case where the simple model is a decision tree.



In this case, there are going to be 2 kinds of parameters P: the weights at each leaf, w, and the number of leaves T in each tree (so that in the above example, T=3 and w=[2, 0.1, -1]).

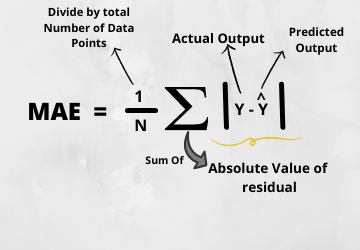
When building a decision tree, a challenge is to decide how to split a current leaf. For instance, in the above image, how could I add another layer to the (age > 15) leaf? A ‘greedy’ way to do this is to consider every possible split on the remaining features (so, gender and occupation), and calculate the new loss for each split; you could then pick the tree which most reduces your loss.

1. **Model performance:**

Model can be evaluated by various metrics such as:

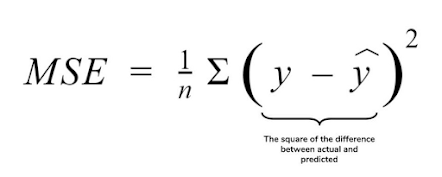
* 1. Mean Absolute Error (MAE)

MAE is a very simple metric which calculates the absolute difference between actual and predicted values.



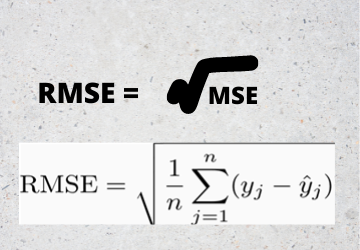
* 1. Mean Squared Error (MSE)

MSE is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value.



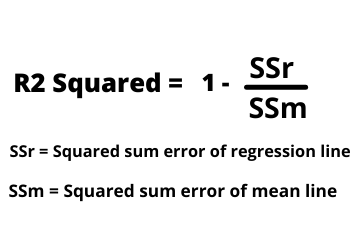
* 1. Root Mean Squared Error (RMSE)

As RMSE is clear by the name itself, that it is a simple square root of mean squared error.



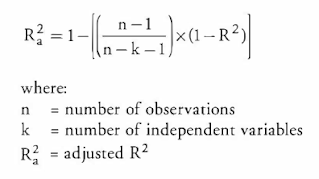
* 1. R Squared (R2)

R2 score is a metric that tells the performance of your model, not the loss in an absolute sense that how many wells did your model perform.



* 1. Adjusted R Squared

The disadvantage of the R2 score is while adding new features in data the R2 score starts increasing or remains constant but it never decreases because it assumes that while adding more data variance of data increases



1. **Hyper parameter tuning**

Hyperparameters are sets of information that are used to control the way of learning an algorithm. Their definitions impact parameters of the models, seen as a way of learning, change from the new hyperparameters. This set of values affects performance, stability and interpretation of a model. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem. Hyperparameters alter the way a model learns to trigger this training algorithm after parameters to generate outputs. We used Grid Search CV

* 1. **Grid Search CV-**Grid Search combines a selection of hyperparameters established by the scientist and runs through all of them to evaluate the model’s performance. Its advantage is that it is a simple technique that will go through all the programmed combinations. The biggest disadvantage is that it traverses a specific region of the parameter space and cannot understand which movement or which region of the space is important to optimize the model

1. **Conclusion:**

That's it! We reached the end of our exercise. Starting with loading the data so far, we have done EDA, null values treatment, encoding of categorical columns, feature selection and then model building. In all of these models our accuracy revolves in the range of 76%

to 99% on train set and 76% to 97% on test set.

So, the performance of our best model which is random forest regressor is 97% which can be said to be good for this large dataset. So, we can deploy this model for solve business problem.

References-

1. MachineLearningMastery

2. GeeksforGeeks

3. Kaggle

4. Towardsdatascience

3. Analytics Vidhya