Name: kancharla.Sairam

Email address: kancharlasairam81@gmail.com

Contact number: 7989881335

Anydesk address: **9515847069**

Years of Work Experience: **7 months**

Date: **15th DEC 2021**

Self Case Study -1: santander customer dissatisfaction problem

"After you have completed the document, please submit it in the classroom in the pdf format."

Please check this video before you get started:
https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg

Overview

*** Write an overview of the case study that you are working on. ***(MINIMUM 200 words)*** ***

Developing a machine learning model with good performance and trusting the model. generally In the banks customer satisfaction is a key measure of success so the best model in our sense is to try to increase the customer satisfaction rate and try to reduce the dissatisfaction rate.

Coming to our business problem this problem is taken from the kaggle competition conducted in the year 2016.

This competition helps to predict whether a customer is dissatisfied with their services . Based on the features provided by the company with the help of our machine learning model we need to predict if the customer is either satisfied or Unsatisfied as early as possible. In case the customer is unsatisfied with banking services then banking management takes proactive steps to improve the customer satisfaction before the customer leaves.

ML formulation of business problem :

This problem is posed in the machine learning classification Model . Target column is dependent column it contains 2 variables {0,1}

-> target value 1 represents the unsatisfied customer

-> target value 0 represents the satisfied customer

we need to develop effecient Machine Learning classification model.

Dataset :

Data link : https://www.kaggle.com/c/santander-customer-satisfaction/data

Dataset containing anonymized numerical variables .

The TARGET column is the variable to predict it equals

- 1 represents unsatisfied customers
- 0 represents satisfied customers

Train data containing 371 columns with target feature

Test data containing 370 columns without target variable.

According to the dataset the task is to predict the probability that each customer in the test set is an unsatisfied customer.

All columns are in numerical format and this is an anonymous dataset we don't have any clue about the features.

Performance metric :

Submissions are evaluated on AUC between the predicted probability and the observed target.

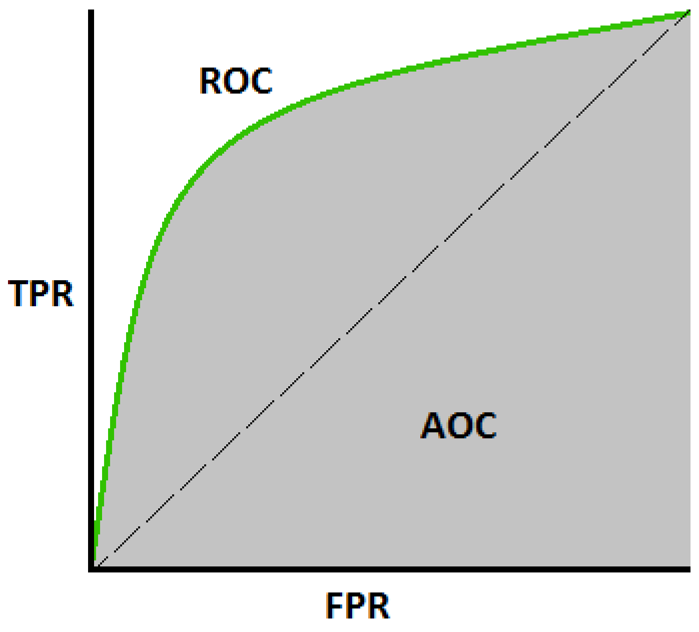
AUC-ROC curve helps us visualize how well our machine learning classifier is performing

AUC-ROC curve is a performance measurement for the classification problems at various threshold values .

Roc (Receiver operating Characteristic) curve is a probability curve & AUC(area under the roc curve) represents the degree of measure of separability . It tells how much a model is capable of distinguishing between classes.

- If AUC value nearer to the 1 which means it has a good measure of separability between classes such type of model is called Good model
- If AUC value is around 0.5 which means the model has no class separability capacity such a model is random model
- If AUC value nearer to the 0 which means it has the worst measure of separability (it is reciprocating the result 1’s as 0’s & 0’s as 1’s) such model is called poor model.

Roc is plotted with TPR against FPR.



Research-Papers/Solutions/Architectures/Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed you can include images or explain with your own diagrams. it is mandatory to write a brief description about that paper. Without understanding of the resource please don't mention it***

1. https://towardsdatascience.com/metrics-for-imbalanced-classification-41c71549bbb5

   From this blog I got some basic sense why we are not using Accuracy as a metric for

   imbalanced dataset

   Incase of imbalanced classes Accuracy can be misleading as high . This metric doesn't show prediction capacity for the minority class.

In this blog they proved experimentally why we are using the AUC-ROC as the metric because it took care of both True positives and False positives other than other metrics.

2. https://www.kaggle.com/solegalli/customer-satisfaction-with-imbalanced-data

In this Kernel drop the constant(variable threshold =0) and quasi constant features & duplicated features.

This data set is imbalanced so perform the SMOTENC technique to balance the dataset

Uses the gradient boosting model

3. https://www.kaggle.com/adarshsng/extensive-advance-feature-selection-tutorial

This kernal tells about the feature selection methods like Step forward feature selection

,Recursive feature selection,Exhaustive feature selection.

4. https://www.kaggle.com/nulikrishna/98-with-pca-randomforest

In this kernel uses the PCA to reduce the dimensionality and uses the Random forest classifier model

5. https://thenewstack.io/3-new-techniques-for-data-dimensionality-reduction-in-machine-learning/

This blog represents the different dimensionality reduction techniques and performs some basic analysis on each technique but according to this blog high correlation filter technique gives a good result other than all techniques.

High Correlation Filter. Data columns with very similar trends are also likely to carry very similar information, and only one of them will be sufficient for classification. Here we calculate the Pearson product-moment correlation coefficient between numeric columns and the Pearson's chi-square value between nominal columns. For the final classification, we only retain one column of each pair of columns whose pairwise correlation exceeds a given threshold. Notice that correlation depends on the column range, and therefore, normalization is required before applying this technique.

SMOTE & SMOTE-NC :

https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

Instead of undersampling, oversampling is better because you keep all the information in the training dataset. But in undersampling we drop a lot of information . Even if this dropped information belongs to the majority class , it is useful information for a modelling algorithm.

The most widely used approach to synthesizing new examples is called the **Synthetic Minority Oversampling TEchnique**.

- SMOTE works by selecting examples that are close in the feature space.
- SMOTE is a technique based on nearest neighbors judged by euclidean distances between data points in the feature space.
- SMOTE-NC is the over sampling technique for Nominal and Continuous features from the imbalanced-learn library and it is a great tool to generate synthetic data to oversample a minority target class in an imbalanced dataset.
- Which creates synthetic data for categorical as well as quantitative features in the dataset.

SMOTE-NC slightly changes the way a new sample is generated by performing something specific for the categorical features . In fact the categories of a new generated sample are decided by picking the most frequent category of the nearest neighbors present during the generation.

NoTE : The data has been split into a training and a testing dataset . It is very important to only apply SMOTE to the training set and not to the testing set to avoid contaminating & introducing biases into the models.

The parameters that can be tuned are K-neighbors . which allows to determine the number of nearest neighbors to create the new sample and sampling strategy , which allows to indicate how many new samples to create .

CORRELATION :

https://machinelearningmastery.com/how-to-calculate-nonparametric-rank-correlation-in-python/

Correlation is a measure of the association between two variables.

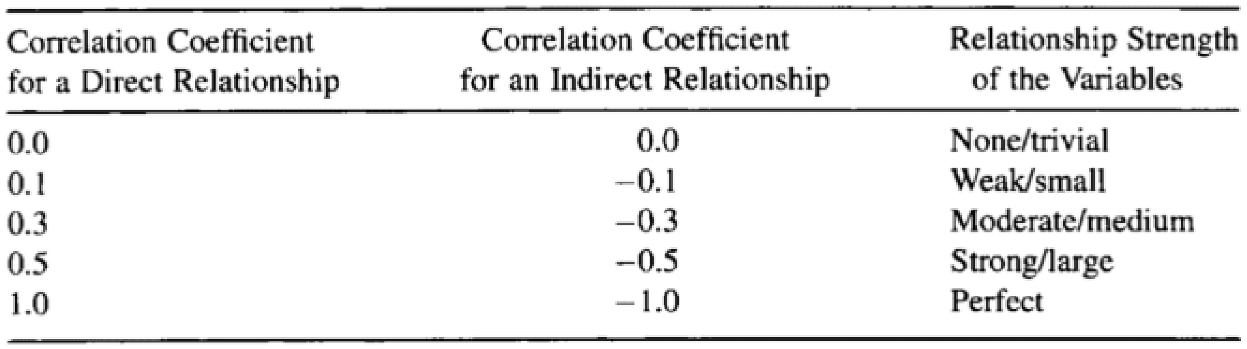
When we do not know the distribution of the variables, we must use nonparametric rank correlation methods.

The variables may have a positive association, meaning that as the values for one variable increase, then values of others also increase.

The variable may have a negative association meaning that as the values of one variable increase, the values of the others decrease.

the association may be neutral, meaning that the variables are not associated.

Correlation values between the -1 to +1.



Correlation Coefficient for a Direct Relationship	Correlation Coefficient for an Indirect Relationship	Relationship Strength of the Variables
0.0	0.0	None/trivial
0.1	−0.1	Weak/small
0.3	−0.3	Moderate/medium
0.5	−0.5	Strong/large
1.0	−1.0	Perfect

Spearman’s rank correlation’s coefficient :

It is a non parametric measure of rank correlation.

It estimates how well the relationship between 2 variables can be described using a “monotonic” function.

Pearson’s correlation estimates “linear” relationships.

Spearman’s correlation estimates “monotonic” relationships.

Kendall's rank correlation estimates “ordinal” relationships.

Kendall’s rank correlation:

The **Kendall rank correlation coefficient**, commonly referred to as **Kendall's τ coefficient** (after the Greek letter τ, tau), is a statistic used to measure the ordinal association between two measured quantities. A **τ test** is a non-parametric hypothesis test for statistical dependence based on the τ coefficient.

The intuition for the test is that it calculates a normalized score for the number of matching or consistent or concordant rankings between the two samples. As such, the test is also referred to as Kendall's concordance test.

Kendall's correlation coefficient is 0.7,which means it is highly correlated.

The p-value is close to zero (and printed as zero), as with the Spearman's test, meaning that we can confidently reject the null hypothesis that the samples are uncorrelated.

PCA:

Principal component analysis is a well known unsupervised dimensionality reduction technique that constructs relevant features/variables through linear (linear PCA) or nonlinear(Kernel PCA) combinations of the original variables(features).

Principal components are the linear combinations of features that have the maximum variance out of all linear combinations.

PCA is particularly useful in processing data where multicollinearity exists between the features

PCA can be used when the dimensions of the input features are high.

PCA can be also used for denoising and data compression.

PCA(principal component analysis) preserves the global shape or structure of the data.

https://datascienceplus.com/principal-component-analysis-pca-with-python/

This blog has complete information about PCA.

TSNE:

t-SNE means t-distribution Stochastic Neighbor Embedding. t-sne is the best dimensionality reduction technique especially for visualization. T-sne preserves local structure

Neighbourhoods are points which are closer to the point xi.

Embedding means picking up a point from high dimensional space and placing that point within the lowest dimensional space.

- Tsne is an iterative algorithm
- t-sne tries to process all the data in iterations and eventually reaches the stage where the clusters are no more moving.
- At every stage tried to move the points and try to improve the embedding
- So it tries to preserve distances as best as possible.
- step_size(no of iterations) is the parameter . step_size increases the stability of structures.
- perplexity (no of data points we get the mess without any structure) so perplexity is < no of data points.
- You have random data then try with all possible perplexities(can't conclude using only one perplexity value, suppose it gives a wrong conclusion then it is extremely dangerous).

Feature engineering

https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/

Forward feature Selection: Forward selection is an iterative method in which we start with having no feature in the model. In each iteration, we keep adding the feature which best improves our model till an addition of a new variable does not improve the performance of the model.

Backward feature Elimination: In backward elimination, we start with all the features and remove the least significant feature at each iteration which improves the performance of the model. We repeat this until no improvement is observed on removal of features.

Recursive Feature elimination: It is a greedy optimization algorithm which aims to find the best performing feature subset. It repeatedly creates models and keeps aside the best or the worst performing feature at each iteration. It constructs the next model with the left features until all the features are exhausted. It then ranks the features based on the order of their elimination.

MODELS

ML algorithm learns a function “f”. Classification is nothing but when you give a new query point using this function says whether it belongs to that class or other class.

LOGISTIC REGRESSION:

Logistic regression is one of the classification techniques and it is a simple & elegant algorithm.

Assumption of logistic regression is if my data is linearly separable / almost linearly separable.

Our main objective is to find the best plane (PI) or line or hyperplane that separates the positive points from the negative points.

For the optimization using the regularizers like L1 , L2 regularizer.

Lambda is the hyperparameter

Best:

- If you have low latency requirements logistic regression works well . suppose we need a low latency system especially with large dimensions. We are using the L1-regularizer.
- If data is almost linearly separable is the best case of logistic regression
- It is good for interpretability and feature importance using weights(coefs).
- Less impact of outliers because of the sigmoid function.
- It is very fast to train, takes O(d) & very easy to train even having large no of data points also.(Before training the data we need to perform feature standardization is mandatory).
- If dimensionality is large then the chance of linear separability is very high . it works fairly very well for large dimensions.

Worst:

- It works badly when data is not linearly separable.
- When data is imbalanced.
- Data having missing values.
- No multiclass classification for the base model. Uses the one vs rest method .
- When multicollinearity exists these models will not work properly.

SUPPORT VECTOR MACHINE:

SVM was a very popular ML algorithm in 1990's for both classification and regression. In svm try to find the plane or hyperplane that separates classes with the maximum margin.

Support vectors are nothing but the points which pass through the Pi+ or Pi- planes; those are called support vectors.

Pi + is the plane parallel to the plane Pi that touches the +ve points.

Pi- is the plane parallel to the plane Pi that touches the -ve points.

Kernelization is the key trick that enables SVM to handle non-linearly separable data sets .

C is the hyperparameter

Best:

- If we can find the right kernel then it works well.
- This can be applied to non-linear problems.
- Interpretability & feature importance is easy for linear SVM..
- the impact of outliers is less.
- If dimensionality is large then SVM works like a charm.

Worst:

- Interpretability and feature importance is hard for kernel SVM.
- If training data is large, training time is high.
- Not used in internet applications . if we have a large number of support vectors(k) then low latency is not possible.

DECISION TREES:

Decision tree algorithms fall under the category of supervised learning. They can be used to solve both regression and classification problems. Decision trees are nested if..else classifier.

Decision trees have high interpretability after being trained . you can literally draw them. Splits the data on the feature which when split previous highest information gain.

Entropy measures uncertainty or impurity or disorder in the given set of values. In other words entropy is a measure of unpredictibility.

By using entropy , gini impurity computes the information gain . based on the information gain we split the features .

Best:

- DT gives interpretability and feature importance.
- Multi class classification is possible

Worst:

- imbalanced data impacts a lot .
- If dimensionality is large then training time is high.
- If you use one-hot encoding then training time will be high
- outliers impact the model.

NAIVE BAYES:

Naive Bayes is a classification algorithm & uses the bayes theorem extensively.

Naive bayes is a conditional probability model. Let's find the probability of a class label (c1,c2,c3,..cK) to the given query point(Xq). P(Ci/Xq)

Here which probability is larger than the other class labels then that given point Xq belongs to that Class label.

NB is the probabilistic Technique

Alpha is the hyper parameter used for the laplace smoothing.

Best cases:

- Naive bayes perform very well on conditional independence of features . assume some features are dependent then Naive Bayes works reasonably well.
- Naive Bayes is the default algorithm when solving text classification problems. NB is very well in text based classification.
- Naive Bayes is extensively used when you have categorical features , rarely used for the real-valued features.

- NB has Great interpretability, feature importance , and low run time.

Worst cases :

- If the conditional independence assumption of naive bayes is false then its performance deteriorates(decreases).
- Naive Bayes is not often used when you have real value features.
- NB Easily gets overfitted if you don't do Laplace smoothing (alpha) correctly. Best alpha can be measured using cross validation.

KNN:

The KNN is a simple, easy to implement supervised machine learning algorithm that can be used to solve both classification and regression problems. KNN is an instance based method.

Find k-nearest points to the new query point Xq . then Take all the class labels of the majority vote. By using a majority vote I can estimate what my class label(Yq) .(Note k must be odd).

K is the hyperparameter.

Best cases :

- If dimensionality (no of features) is low then this works best.
- If you know the right distance measure then KNN is a good option.
- Distance measures are euclidean distance,manhattan distance,Minkowski distance, Hamming distance , cosine distence. By using these distances pick the k-nearest neighbours.

Worst cases:

- If dimensionality is large then this may face the curse of dimensionality problem & takes more run time and less interpretability.
- When you want a low latency product then KNN is not a good option.
- If query points are far away from points in the dataset then we can't classify them exactly.
- For the jumbled dataset points , for randomly spreading data points, mixed data points can't be classified exactly.

ENSEMBLES:

Ensemble means a group or collection of things . In ML perspective multiple base models are used together M = {m1,m2,m3,m4....m$_k$} . ensembles are basically 4 types bagging, boosting,stacking,cascading.

BOOSTING:

An ensemble learning strategy that trains a series of weak models , each one attempting to correctly predict the observations the previous model got wrong.

Boosting is actually an ensemble of learning algorithms which combines the prediction of several base estimators in order to improve robustness over a single estimator.

Boosting = low variance ,high bias base models + additively combine{reduce bias while keep our variance as low}.

GRADIENT BOOSTING:

Gradient Boosting algorithm used when we deal with plenty of data to make a prediction with high prediction power.

It combines multiple weak or average predictors to build a strong predictor.

XGBOOST:

The xgboost has an immensely high predictive power which makes it the best choice for accuracy in events as it possesses both linear model and the tree learning algorithm, making the algorithm almost 10X faster than existing gradient boosting techniques.

XG boost is also a regularized boosting technique. This helps to reduce overfit modelling and has a massive support for languages such as scala,Java,R,python,c++.

- Xg boost stands for eXtreme Gradient Boosting.
- Xg boost is an advanced implementation of gradient boosting. This algorithm has high predictive power and is ten times faster than any other gradient boosting techniques.

LiGHT GBM:

LightGBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages.

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Parallel and GPU learning supported.
- Capable of handling large scale data

RANDOM FOREST CLASSIFIER:

Random forest is a learning method that operates by constructing multiple decision trees. The final decision is made based on the majority of the trees and is chosen by the random forest.

Random Forest classifier that contains a no of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of the dataset.

The greater accuracy of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Each tree gets a random sample of observations with replacement. Each tree gets all features , but at each node only a subset of those features are available.

Random Forest is a tree-based machine learning algorithm that leverages the power of multiple decision trees for making decisions.

Bagging or bootstrap averaging is a technique where multiple models are created on the subset of data, and the final predictions are determined by combining the predictions of the models.

- Random-forest is the most popular bagging technique.
- Random sampling means Bootstrap Aggregation with replacement.
- Forest means group of Decision Trees with reasonable depth.
- Using Decision Tree as a base learner.
- Doing row sampling with replacement & Doing column sampling(feature sampling)
- Then Doing Aggregation.

Best cases:

- Random Forest is based on the bagging algorithm and uses Ensemble learning technique. It creates as many trees on the subset of the data and combines the output of all the trees. In this way it reduces the variance and therefore improves the accuracy.
- RF works well with both categorical and continuous variables.
- The RF algorithm is very stable. Even if a new dataset is introduced in the dataset, the overall algorithm is not affected much since the new data may impact the tree ,but it is very hard for it to impact all the trees.
- RF is usually robust to outliers and can handle them automatically.
- No feature scaling(standardization & Normalization) required in case of RF as it uses rule based approach instead of distance calculations.

Worst Cases:

- RF requires much more time to train as compared to decision trees as it generates a lot of trees and makes decisions on the majority of votes.
- RF requires much more computational power and resources .

HYPERPARAMETERTUNING

For finding the best value of hyperparameters we are using 2 approaches broadly Grid Search CV , Random Search CV.

https://analyticsindiamag.com/top-8-approaches-for-tuning-hyperparameters-of-machine-learning-models/

Random searched cv:

Random search cv means randomly picking the values in the given interval or set of values. At every point find the cross validation error, draw the plot & find which one is better for classification.

Random search cv is almost as good as grid search . it is faster than grid search sometimes and no of hyperparameters are large then random search cv is better than grid search cv.

Grid search cv:

Grid search is a basic method for hyperparameter tuning. It performs an exhaustive search on the hyperparameter set specified by users. This approach is the most straightforward leading to the most accurate predictions. Using this tuning method, users can find the optimal combination. Grid search is applicable for several hyper-parameters, however, with limited search space.

First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

*** When you are doing the basic EDA and building the First Cut Approach you should not refer any blogs or papers ***

1. I want to load the dataset and perform some basic analysis to understanding about the given dataset like dataset information , shape, describe, no of columns in the dataset and data type of those features .. etc
2. Next I need to identify the dependent /target feature.
3. Check the dataset is balanced or imbalanced based on the target value
4. If the dataset is imbalanced try to balance the dataset by performing the undersampling or over sampling techniques (SMOTE & SMOTENC).
5. Check the missing values in the dataset and try to fill those missing values by using imputation techniques or ignoring those rows(datapoints).
6. Check the outliers in the dataset using local reachability density, local outlier factor , reachability distance(A,B) ...etc. And some plots are also useful to identify the outliers in the dataset.

7. In this dataset is an imbalanced dataset and all are numeric columns.
8. All are the anonymized features so hard to interpret the features
9. Remove the constant features , duplicate features , find the correlation of each feature with the target variable .
10. correlation techniques are pearson correlation coefficient, spearman rank correlation coefficient,kendall correlation coefficient.
11. If the feature has high correlation with the target feature those features are useful in the classification.
12. Dataset contains 371 features with target variable
13. To understand the data for the visualization using TSNE.
14. By performing the feature engineering & dimensionality reduction techniques I tried to reduce the features.
15. For feature engineering I need to perform techniques like forward feature selection, backward feature selection .
16. Then try to build a Machine Learning model using train data.
17. I want to use all the ML classification models

    Decision trees , Naive Bayes, Gradient boosting , Xgboost , Random forest classifier .. etc

18. Perform the hyperparameter tuning to find the best model with best value.
19. Evaluate each model with AUC score .
20. Finally choose which model gives the best score value.
21. By using that model predict the test data output values.
22. Create the sample submission data and submit in the kaggle competition.

Notes when you build your final notebook:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple StandardScalar
2. You should not read train data files

3. The function1 takes only one argument “X” (a single data points i.e 1*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
    a. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
    b. so in your final notebook, you need to pass only those two values
    c. def final(X):

```
        preprocess data i.e data cleaning, filling missing values etc
        compute features based on this X
        use pre trained model
        return predicted outputs
final([time, location])
```

    d. in the instructions, we have mentioned two functions one with original values and one without it
    e. final([time, location])   # in this function you need to return the predictions, no need to compute the metric
    f. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y_predict)
4. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data
5. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible
6. Check this live session: https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models