Hotel Profitability-Machine Learning Study

Submitted as part of the requirements for

CE-802: Machine Learning and Data Mining

Name: Siddharth Shyamsunder

Lecturer: Dr. Luca Citi

Submission Date: 16th January 2019

Pilot Study Word Count:664

Comparative Study Word Count: 893

Contents

Pilot Study: 3

- Outline: 3
- Study: 3

Comparative Study: 4

Appendix: 8

Pilot Study:

Outline:

We have been presented an area of study and decision making, where in, as a Machine Learning Consultant, we are approached by a Hotel Manager of a large brand of hotels, who wishes to realize the feasibility of whether upon building a hotel at a particular area, would such a hotel be profitable or not. Given the role of making this decision, the Hotel Manager has provided us with past information of profitability on data on the basis of Socio Economic information based on the location, demography, and certain other conditions.

Study:

Considering the case of study is upon decision making, and such decision making skills will be arbitrarily infeasible to conduct on a set of related raw data using standard means of algorithms. I t is safe to assume that the said problem will come under theidea of a Machine Learning Problem.

Once we have considered the fact that this problem is a Machine Learning Problem, we would further categorize this problem on the basis of Supervised, Unsupervised or Reinforcement Learning. As in this case we will need to make a decision on the standard set of data rather than a sequence of steps to realize the output. It is safe to rule out Reinforcement Learning from the picture.

Also as the Hotel Manager has provided past information on the features that enhance or break profitability of building a hotel in a particular location. We have a clear picture of what kind of data has been provided, this problem would come under the Supervised Learning algorithm, and we can rule out Unsupervised Learning.

Under Supervised Learning Algorithm, as we have to decide whether the problem will result in a profitability or a loss of the building of the hotel, at a particular location, this will come under **Classification** algorithms.

When considering some of the features that will decide whether building a hotel would be profitable at a particular location, one should look into various factors, like population density, percentage of tourism of the region, distance to nearby commercial ares in order to purchase food, blankets and so on. Percentage of affordability of the hotel, and other salient features, that could be region specific or as per the terms and policies of the hotel.

As we talked about various features that will play a certain role in the profitability of the hotel. The raw data that the manager provides may also contain many unwanted attributes, which may not provide any gain to the decision making process, so in order to make the process more convincing we will employ considering only informative attributes through the means of **Decision Trees**. Further to give a better output and to avoid overfitting of data. We will perform pruning of the information in the decision tree. Also as there are a lot of features involved in the process of finding the solution of profitability.

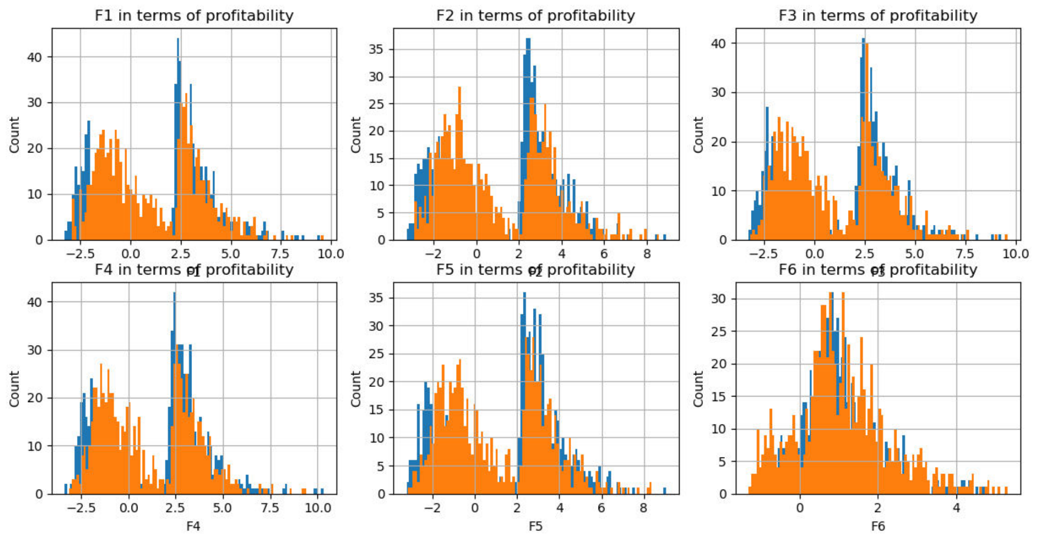
Combining all such features would create a very strong labeled cluster between the 2 classes. Hence it will be best to employ an instance-based Learning Method like the kNN Model. Further more, a clear gap or margin will be needed to properly distinguish profitable Hotels from Loss Hotels, so we will employ the use of **Support Vector Classification** Algorithm.

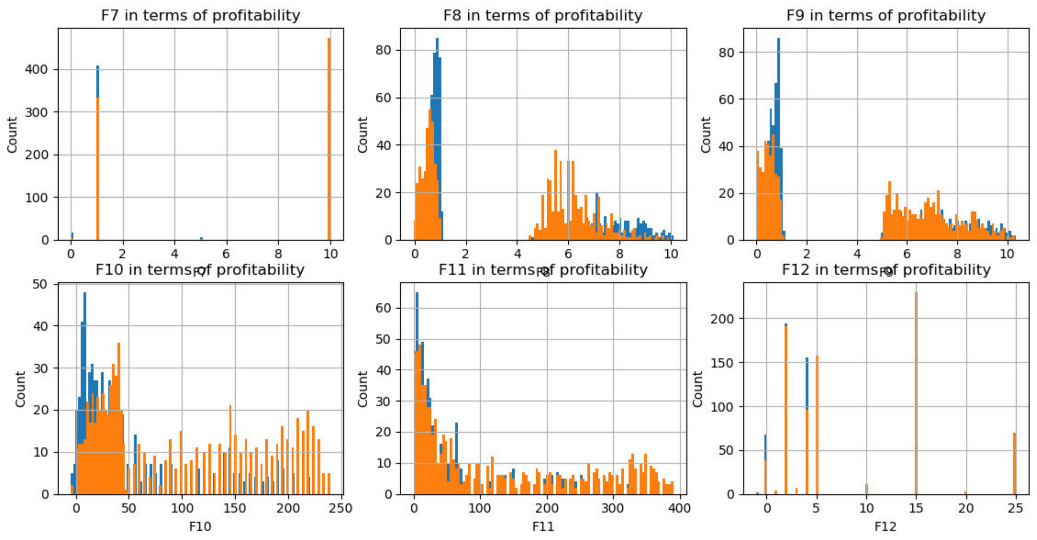
In order to deploy this algorithm, we will need to first normalize the data to get a clear definite range of the numerical data. After which we will perform, cross validation wherein we take the entire dataset and create it into blocks such that at every stage, 5 data records will be used as the test set and the remaining data records will be the training records. Then we will perform the Machine Learning Algorithms discussed on each of these pairs and get an accuracy standard for each pair, which upon averaging, we will get the efficiency of the algorithm. Then we will ensure the most accurate algorithm be used to predict the values to generate the output for the actual test set.

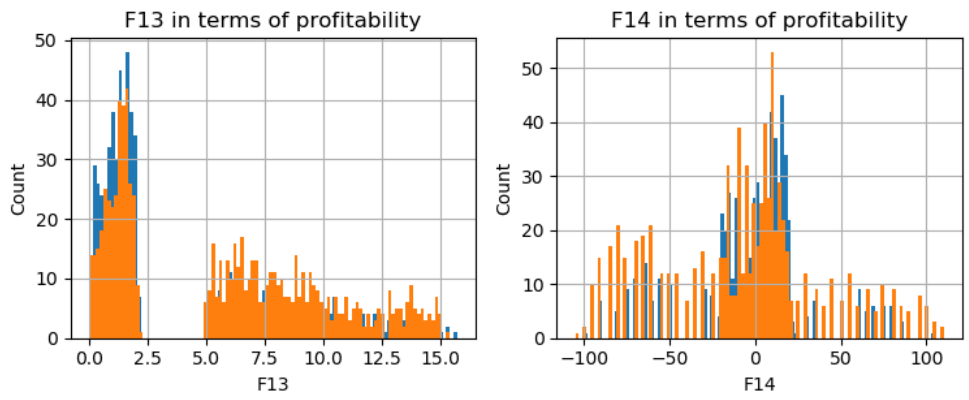
Comparative Study:

Upon receiving the data from the Manager, we realized that as there are 14 different type of features, it is possible that some of the features might be irrelevant. And the values thst we received were ambiguous in structure.

Hence, upon first receiving the data, we performed some explorative analysis on the data though means of histograms

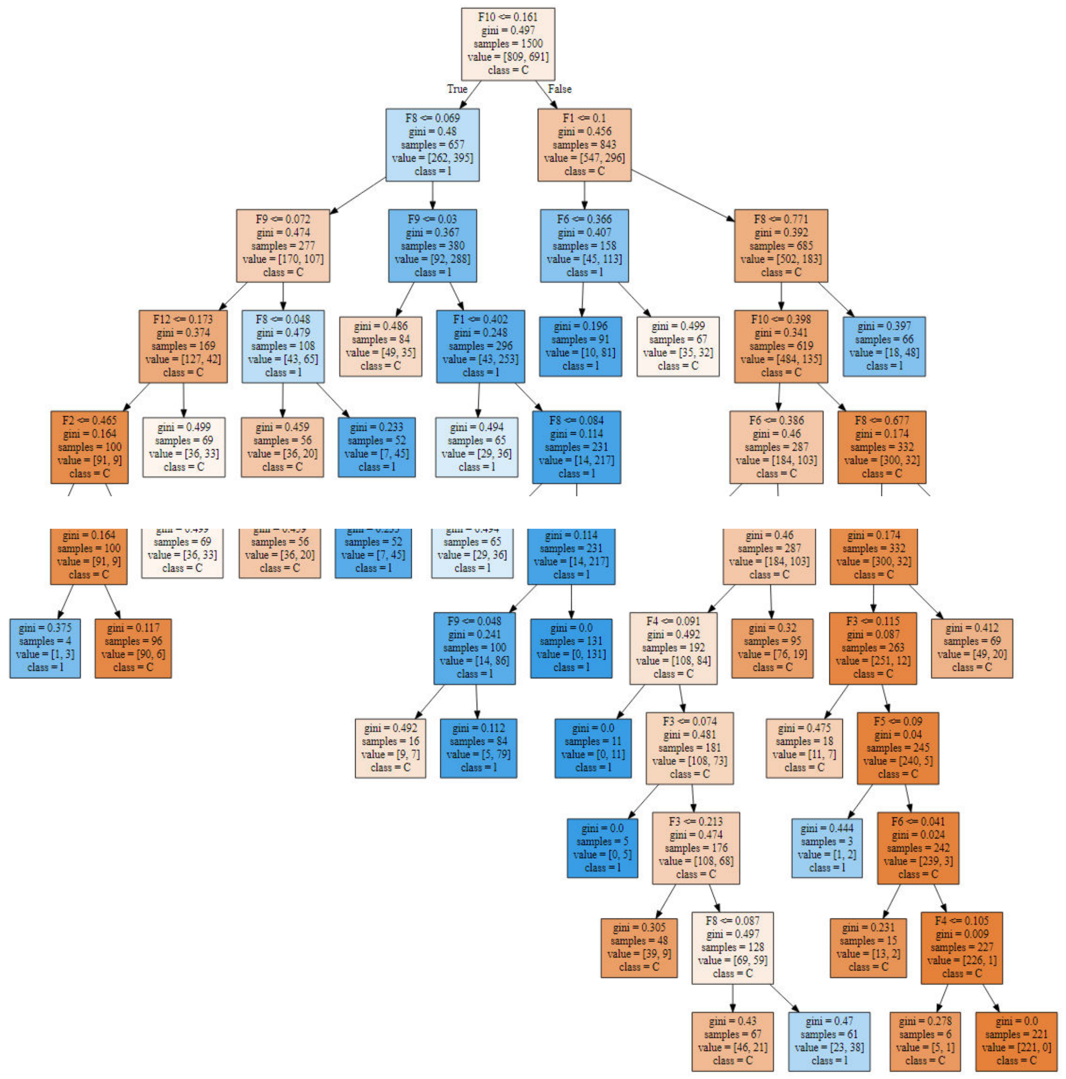






From the above Histograms, we can note that the Red color signals the feature will be profitable, and the Blue Color signals that the hotel would not be profitable. Also we can notice, that there is a lot of bias between the profitabledata counts and the non profitable counts, and there is a lot of range difference that makes it hard to accumulate the features. So the first step which was carried out was normalization of the data for each feature, wherein we specified the range of each feature to lie between 0 and 1.

Once the data was normalized, we carried out 3 Machine Learning Algorithms on the data. As the data consisted of a lot of random discrete values under the features, we figured there would be a cause of overfitting if in case a Decision Tree would be made, so we splitted the samples into 100 blocks and performed a Pruned Decision Tree algorithm on the data.



F10 <= 0.161 gini = 0.497 samples = 1500 value = [809, 691] class = C
True
False
F8 <= 0.069 gini = 0.48 samples = 657 value = [262, 395] class = 1
F1 <= 0.1 gini = 0.456 samples = 843 value = [547, 296] class = C
F9 <= 0.072 gini = 0.474 samples = 277 value = [170, 107] class = C
F9 <= 0.03 gini = 0.367 samples = 380 value = [92, 288] class = 1
F6 <= 0.366 gini = 0.407 samples = 158 value = [45, 113] class = 1
F8 <= 0.771 gini = 0.392 samples = 685 value = [502, 183] class = C
F12 <= 0.173 gini = 0.374 samples = 169 value = [127, 42] class = C
F8 <= 0.048 gini = 0.479 samples = 108 value = [43, 65] class = 1
gini = 0.486 samples = 84 value = [49, 35] class = C
F1 <= 0.402 gini = 0.248 samples = 296 value = [43, 253] class = 1
gini = 0.196 samples = 91 value = [10, 81] class = 1
gini = 0.499 samples = 67 value = [35, 32] class = C
F10 <= 0.398 gini = 0.341 samples = 619 value = [484, 135] class = C
gini = 0.397 samples = 66 value = [18, 48] class = 1
F2 <= 0.465 gini = 0.164 samples = 100 value = [91, 9] class = C
gini = 0.499 samples = 69 value = [36, 33] class = C
gini = 0.459 samples = 56 value = [36, 20] class = C
gini = 0.233 samples = 52 value = [7, 45] class = 1
gini = 0.494 samples = 65 value = [29, 36] class = 1
F8 <= 0.084 gini = 0.114 samples = 231 value = [14, 217] class = 1
F6 <= 0.386 gini = 0.46 samples = 287 value = [184, 103] class = C
F8 <= 0.677 gini = 0.174 samples = 332 value = [300, 32] class = C
gini = 0.164 samples = 100 value = [91, 9] class = C
samples = 69 value = [36, 33] class = C
samples = 56 value = [36, 20] class = C
samples = 52 value = [7, 45] class = 1
samples = 65 value = [29, 36] class = 1
gini = 0.114 samples = 231 value = [14, 217] class = 1
gini = 0.46 samples = 287 value = [184, 103] class = C
gini = 0.174 samples = 332 value = [300, 32] class = C
gini = 0.375 samples = 4 value = [1, 3] class = 1
gini = 0.117 samples = 96 value = [90, 6] class = C
F9 <= 0.048 gini = 0.241 samples = 100 value = [14, 86] class = 1
gini = 0.0 samples = 131 value = [0, 131] class = 1
F4 <= 0.091 gini = 0.492 samples = 192 value = [108, 84] class = C
gini = 0.32 samples = 95 value = [76, 19] class = C
F3 <= 0.115 gini = 0.087 samples = 263 value = [251, 12] class = C
gini = 0.412 samples = 69 value = [49, 20] class = C
gini = 0.492 samples = 16 value = [9, 7] class = C
gini = 0.112 samples = 84 value = [5, 79] class = 1
gini = 0.0 samples = 11 value = [0, 11] class = 1
F3 <= 0.074 gini = 0.481 samples = 181 value = [108, 73] class = C
gini = 0.475 samples = 18 value = [11, 7] class = C
F5 <= 0.09 gini = 0.04 samples = 245 value = [240, 5] class = C
gini = 0.0 samples = 5 value = [0, 5] class = 1
F3 <= 0.213 gini = 0.474 samples = 176 value = [108, 68] class = C
gini = 0.444 samples = 3 value = [1, 2] class = 1
F6 <= 0.041 gini = 0.024 samples = 242 value = [239, 3] class = C
gini = 0.305 samples = 48 value = [39, 9] class = C
F8 <= 0.087 gini = 0.497 samples = 128 value = [69, 59] class = C
gini = 0.231 samples = 15 value = [13, 2] class = C
F4 <= 0.105 gini = 0.009 samples = 227 value = [226, 1] class = C
gini = 0.43 samples = 67 value = [46, 21] class = C
gini = 0.47 samples = 61 value = [23, 38] class = 1
gini = 0.278 samples = 6 value = [5, 1] class = C
gini = 0.0 samples = 221 value = [221, 0] class = C

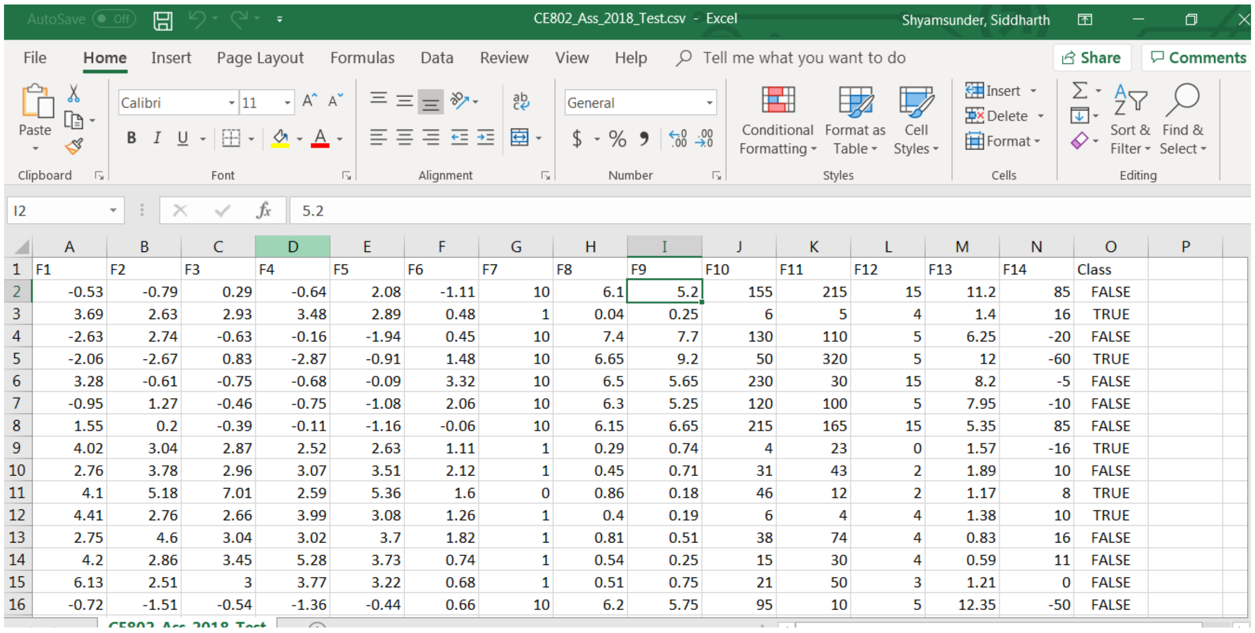
From the pruned Decision tree, we areable to gather certain aspects:

As an initial point of view, the features F7,F11,F13, F14 are not very informative and so they were not labeled in the decision tree, the colors of the decision tree specify the type of class the data would eventually reach, tree size is much smaller, as a result of pruning, and the accuracy observed was: 72.93.

Some of the reasons which allowed us to choosea decision tree, was its simplicity in handling data, and one of thefew machine Learning Algortihms that can be properly visualized, in our case, we made use of graphviz package to design the above decision tree, and, decision tree being known for its versatility in handling both numerical and categorical data, was a strong advantage of choosing the decision tree. Also a decision tree primarily can handle raw data, and we only performed normalization to our dataset as predominantly our dataset consisted of only numerical values of a mixed range.

Although decision tree has its positive sides, so does it also have itsnegstive points, in our case we solved one of the biggest disadvantages of the decision tree, which was the case of overfitting of data. We solved it through means of pruning, or rather collecting of a range of samples, rather than making use of a new subtree for each of the data values. Also decision trees can be more biased in terms of neutralization of categorization of data.

Then as we have a high number of features, we experimented on the dataset using thr Support Vector Mechanism(SVM). Herein, upon using the SVM method, as the SVM automatically uses the subsert of the data to determine the decision, this coupled with prior normalization and cross validation proved to give an effectively high accuracy rate of 74.73%. Which being the highest in terms of performance for our 3 methods was further used to predict the profitability of the unknown test dataset, as shown below.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	Class
2	-0.53	-0.79	0.29	-0.64	2.08	-1.11	10	6.1	5.2	155	215	15	11.2	85	FALSE
3	3.69	2.63	2.93	3.48	2.89	0.48	1	0.04	0.25	6	5	4	1.4	16	TRUE
4	-2.63	2.74	-0.63	-0.16	-1.94	0.45	10	7.4	7.7	130	110	5	6.25	-20	FALSE
5	-2.06	-2.67	0.83	-2.87	-0.91	1.48	10	6.65	9.2	50	320	5	12	-60	TRUE
6	3.28	-0.61	-0.75	-0.68	-0.09	3.32	10	6.5	5.65	230	30	15	8.2	-5	FALSE
7	-0.95	1.27	-0.46	-0.75	-1.08	2.06	10	6.3	5.25	120	100	5	7.95	-10	FALSE
8	1.55	0.2	-0.39	-0.11	-1.16	-0.06	10	6.15	6.65	215	165	15	5.35	85	FALSE
9	4.02	3.04	2.87	2.52	2.63	1.11	1	0.29	0.74	4	23	0	1.57	-16	TRUE
10	2.76	3.78	2.96	3.07	3.51	2.12	1	0.45	0.71	31	43	2	1.89	10	FALSE
11	4.1	5.18	7.01	2.59	5.36	1.6	0	0.86	0.18	46	12	2	1.17	8	TRUE
12	4.41	2.76	2.66	3.99	3.08	1.26	1	0.4	0.19	6	4	4	1.38	10	TRUE
13	2.75	4.6	3.04	3.02	3.7	1.82	1	0.81	0.51	38	74	4	0.83	16	FALSE
14	4.2	2.86	3.45	5.28	3.73	0.74	1	0.54	0.25	15	30	4	0.59	11	FALSE
15	6.13	2.51	3	3.77	3.22	0.68	1	0.51	0.75	21	50	3	1.21	0	FALSE
16	-0.72	-1.51	-0.54	-1.36	-0.44	0.66	10	6.2	5.75	95	10	5	12.35	-50	FALSE

Also as the feature set is much lesser than the set of samples, it is optimum to make use of the multiple versatile nature of the SVM, to create an optimum marginality factor C to differentiate both the classes using this algorithm.

We discussed most of the advantages of using SVM as this algorithm performed the most accurate of our 3 Learning Algorithms. However there are some disadvantages with SVM, like SVM method can only handle datasets wherein the number of features are much lesser than the number of samples which was true in our case as we had 1500 samples and only 14 features. Plus in addition to this, there was much required importance of making use of cross validation in 5 Fold to handle the dataset.

As for the 3rd Machine Learning Example. Once again as a similar condition to instance based Learning, we made use of Instance-Based Learning to develop a 3NN scatter plor Machine Learning Algorith. Though unlike the Support Vector Functionality, as we do not make use of marginality or kernels, here we consider finding common attributes between features associated with a new training example and its neighbors to help classify the data into the respective class on the basis of plurality of votes from the neighbors.

Here once again, we initially normalized our training dataset by ranging the values between 0 and one, then we ran the algorithm by making use of cross validation as the means of splitting the data into training sets and test sets and performing cross validation. And upon averaging the crossvalidation scores we were able to get an average accuracy of about 72.33%.

Some of the major advantages of using the kNN Model was its simplicity of training data and the versatility it holdsin handling different types of data.

However unlike the SVM. KNN does not look into border marginality analysis, and it is difficult totest unseen data, which we reduced the complexity, by working around with the data by normalizing the data.

Below are the performance statistics of the 3 Machine Learning methods.

S.No	Machine Learning Method	Accuracy
1	Pruned Decision Tree	72.93(+/-2)%
2	Support Vector Machine	74.73 (+/-3)%
3	3NN Instance Based Method	72.33 (+/-2)%

Appendix:

Attached is the Jupyter Notebook version of the code that we have provided.

```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import graphviz\n",
    "import numpy as np\n",
    "from sklearn import tree\n",
    "from sklearn.tree import DecisionTreeClassifier\n",
    "from sklearn.model_selection import train_test_split\n",
    "from matplotlib import pyplot as plt\n",
    "import seaborn as sns\n",
    "from sklearn.metrics import accuracy_score\n",
    "from sklearn import svm,preprocessing\n",
    "from sklearn.model_selection import KFold, cross_val_score\n",
    "from sklearn.neighbors import KNeighborsClassifier\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "data=pd.read_csv('CE802_Ass_2018_Data.csv')\n",
    "print(data.info())"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
"metadata": {},

"outputs": [],

"source": [

 "posf1=data[data['Class']==True]['F1']\n",

 "negf1=data[data['Class']==False]['F1']\n",

 "posf2=data[data['Class']==True]['F2']\n",

 "negf2=data[data['Class']==False]['F2']\n",

 "posf3=data[data['Class']==True]['F3']\n",

 "negf3=data[data['Class']==False]['F3']\n",

 "posf4=data[data['Class']==True]['F4']\n",

 "negf4=data[data['Class']==False]['F4']\n",

 "posf5=data[data['Class']==True]['F5']\n",

 "negf5=data[data['Class']==False]['F5']\n",

 "posf6=data[data['Class']==True]['F6']\n",

 "negf6=data[data['Class']==False]['F6']\n",

 "posf7=data[data['Class']==True]['F7']\n",

 "negf7=data[data['Class']==False]['F7']\n",

 "posf8=data[data['Class']==True]['F8']\n",

 "negf8=data[data['Class']==False]['F8']\n",

 "posf9=data[data['Class']==True]['F9']\n",

 "negf9=data[data['Class']==False]['F9']\n",

 "posf10=data[data['Class']==True]['F10']\n",

 "negf10=data[data['Class']==False]['F10']\n",

 "posf11=data[data['Class']==True]['F11']\n",

 "negf11=data[data['Class']==False]['F11']\n",
  "posf12=data[data['Class']==True]['F12']\n",
  "negf12=data[data['Class']==False]['F12']\n",
  "posf13=data[data['Class']==True]['F13']\n",
  "negf13=data[data['Class']==False]['F13']\n",
  "posf14=data[data['Class']==True]['F14']\n",
  "negf14=data[data['Class']==False]['F14']\n"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "fig=plt.figure(figsize=(100,100))\n",
  "ax=fig.add_subplot(2,3,1)\n",
  "ax.set_xlabel('F1')\n",
  "ax.set_ylabel('Count')\n",
  "plt.title('F1 in terms of profitability')\n",
  "posf1.hist(bins=100,label='Positive')\n",
  "negf1.hist(bins=100,label='Negative')\n",
  "\n",
  "ax2=fig.add_subplot(2,3,2)\n",
  "ax2.set_xlabel('F2')\n",
  "ax2.set_ylabel('Count')\n",
"plt.title('F2 in terms of profitability')\n",

"posf2.hist(bins=100,label='Positive')\n",

"negf2.hist(bins=100,label='Negative')\n",

"\n",

"ax3=fig.add_subplot(2,3,3)\n",

"ax3.set_xlabel('F3')\n",

"ax3.set_ylabel('Count')\n",

"plt.title('F3 in terms of profitability')\n",

"posf3.hist(bins=100,label='Positive')\n",

"negf3.hist(bins=100,label='Negative')\n",

"\n",

"ax4=fig.add_subplot(2,3,4)\n",

"ax4.set_xlabel('F4')\n",

"ax4.set_ylabel('Count')\n",

"plt.title('F4 in terms of profitability')\n",

"posf4.hist(bins=100,label='Positive')\n",

"negf4.hist(bins=100,label='Negative')\n",

"\n",

"ax5=fig.add_subplot(2,3,5)\n",

"ax5.set_xlabel('F5')\n",

"ax5.set_ylabel('Count')\n",

"plt.title('F5 in terms of profitability')\n",

"posf5.hist(bins=100,label='Positive')\n",

"negf5.hist(bins=100,label='Negative')\n",

"\n",
   "ax6=fig.add_subplot(2,3,6)\n",
   "ax6.set_xlabel('F6')\n",
   "ax6.set_ylabel('Count')\n",
   "plt.title('F6 in terms of profitability')\n",
   "posf6.hist(bins=100,label='Positive')\n",
   "negf6.hist(bins=100,label='Negative')\n",
   "plt.show()\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": [
   "\n",
   "fig2=plt.figure(figsize=(100,100))\n",
   "\n",
   "ax7=fig2.add_subplot(2,3,1)\n",
   "ax7.set_xlabel('F7')\n",
   "ax7.set_ylabel('Count')\n",
   "plt.title('F7 in terms of profitability')\n",
   "posf7.hist(bins=100,label='Positive')\n",
   "negf7.hist(bins=100,label='Negative')\n",
   "\n",
"ax8=fig2.add_subplot(2,3,2)\n",

"ax8.set_xlabel('F8')\n",

"ax8.set_ylabel('Count')\n",

"plt.title('F8 in terms of profitability')\n",

"posf8.hist(bins=100,label='Positive')\n",

"negf8.hist(bins=100,label='Negative')\n",

"\n",

"ax9=fig2.add_subplot(2,3,3)\n",

"ax9.set_xlabel('F9')\n",

"ax9.set_ylabel('Count')\n",

"plt.title('F9 in terms of profitability')\n",

"posf9.hist(bins=100,label='Positive')\n",

"negf9.hist(bins=100,label='Negative')\n",

"\n",

"ax10=fig2.add_subplot(2,3,4)\n",

"ax10.set_xlabel('F10')\n",

"ax10.set_ylabel('Count')\n",

"plt.title('F10 in terms of profitability')\n",

"posf10.hist(bins=100,label='Positive')\n",

"negf10.hist(bins=100,label='Negative')\n",

"\n",

"ax11=fig2.add_subplot(2,3,5)\n",

"ax11.set_xlabel('F11')\n",

"ax11.set_ylabel('Count')\n",

"plt.title('F11 in terms of profitability')\n",
    "posf11.hist(bins=100,label='Positive')\n",
    "negf11.hist(bins=100,label='Negative')\n",
    "\n",
    "ax12=fig2.add_subplot(2,3,6)\n",
    "ax12.set_xlabel('F12')\n",
    "ax12.set_ylabel('Count')\n",
    "plt.title('F12 in terms of profitability')\n",
    "posf12.hist(bins=100,label='Positive')\n",
    "negf12.hist(bins=100,label='Negative')\n",
    "\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "fig3=plt.figure(figsize=(100,100))\n",
    "\n",
    "ax13=fig3.add_subplot(2,3,1)\n",
    "ax13.set_xlabel('F13')\n",
    "ax13.set_ylabel('Count')\n",
    "plt.title('F13 in terms of profitability')\n",
    "posf13.hist(bins=100,label='Positive')\n",
    "negf13.hist(bins=100,label='Negative')\n",
    "\n",
    "ax14=fig3.add_subplot(2,3,2)\n",
    "ax14.set_xlabel('F14')\n",
    "ax14.set_ylabel('Count')\n",
    "plt.title('F14 in terms of profitability')\n",
    "posf14.hist(bins=100,label='Positive')\n",
    "negf14.hist(bins=100,label='Negative')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "features=['F1','F2','F3','F4','F5','F5','F6','F7','F8','F9','F10','F11','F12','F13','F14']\n",
    "x=data[features]\n",
    "minmaxscaler=preprocessing.MinMaxScaler(feature_range=(0,1))\n",
    "x=minmaxscaler.fit_transform(x)\n",
    "y=data['Class']\n",
    "clf=DecisionTreeClassifier(min_samples_split=100)\n",
    "dt=clf.fit(x,y)\n",
    "\n",
```

"ddt=tree.export_graphviz(clf,out_file='tree.dot',feature_names=features,class_names='Class',filled=True)\n",

"graph=graphviz.Source(ddt)\n",

"\n",

"scores = cross_val_score(clf, x, y, cv=5)\n",

"print(\"Accuracy of a Decision Tree is:\",round(np.mean((scores*100)),2),\"+/-\",round(np.std(scores*100),0))\n"

]

},

{

"cell_type": "code",

"execution_count": null,

"metadata": {

"scrolled": true

},

"outputs": [],

"source": [

"clf2 = svm.SVC(gamma=0.01, C=100.)\n",

"dtf=data[features]\n",

"minmaxscaler=preprocessing.MinMaxScaler(feature_range=(0,1))\n",

"dtf=minmaxscaler.fit_transform(dtf)\n",

"dtc=data['Class']\n",

"print(dtf,'\\n\\n')\n",

"print(dtc,'\\n\\n')\n",

"clf2=clf2.fit(dtf,dtc)\n",

"scores = cross_val_score(clf2, dtf, dtc, cv=5)\n", "print(\"Accuracy of a Support Vector Machine is:\",round(np.mean((scores*100)),2),\"+/-\",round(np.atd(scores*100),0))\n"

]

},

{

"cell_type": "code",

"execution_count": null,

"metadata": {

"scrolled": false

},

"outputs": [],

"source": [

"clf3=KNeighborsClassifier(n_neighbors=3)\n",

"clf3.fit(dtf,dtc)\n",

"scores = cross_val_score(clf3, dtf, dtc, cv=5)\n",

"print(\"Accuracy of a 3NN Instance based Learning is:\",round(np.mean((scores*100)),2),\"+/-\",round(np.std(scores*100),0))\n",

"\n",

"data2=pd.read_csv('CE802_Ass_2018_Test.csv')\n",

"x1=data2[features]\n",

"minmaxscaler=preprocessing.MinMaxScaler(feature_range=(0,1))\n",

"x1=minmaxscaler.fit_transform(x1)\n",

"y1=data2['Class']\n",

"op1=clf2.predict(x1)\n",

"print(op1)\n",

"x1=data2[features]\n",

```
   "df=pd.DataFrame(op1)\n",
   "df.to_csv('Try.csv')\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": []
 }
],
"metadata": {
 "kernelspec": {
  "display_name": "Python 3",
  "language": "python",
  "name": "python3"
 }
},
"nbformat": 4,
"nbformat_minor": 2
}
```