Machine Learning Engineer Nanodegree

Capstone Project

Perihane Youssef Yasser

February 6, 2019

I. Definition

Project Overview

Day nurseries are a great option if you want your child's learning and care to have structure. Staff are trained to create a safe, happy and stimulating environment where your child can play and develop. Your child will be able to do a wide variety of activities at day nursery. These activities will be designed to encourage your child's social, creative, communication and listening skills and his physical development[1].

The problem lies in choosing the best nursery for your child, where he can make use of his potentials and skills and feel accepted among his peers. That's why nurseries put such great effort in interviewing new children and set very strict criteria on evaluating and accepting them depending on the child and his parents financial and social standings and what they seek in a "day care" facility for their child.

The dataset used for this project is "Nursery Data Set", from UCI Machine Learning Repository[2]. It was published in 1997 and it has 12960 rows and 8 columns, each column represent one feature of a child and his parents (a piece of information). Each row represents a data point: one child observation. The target variable to be predicted in this problem is the "NURSERY" variable. It contains four values "not recommended, recommended, very recommended, priority and special priority". The rest of columns represent information about occupation of parents and child's nursery, family structure and financial standing, and social and health picture of the family. All the columns contains categorical variables and they will be encoded to numerical values to be able to feed them to different machine learning models, and there are no missing values.

Problem Statement

Our main concern is to decide if a child's application in the nursery will be accepted or rejected. Since that any poor choice will have a negative effect on the child and his colleagues as children are so sensitive in their first years, during which learning depends mainly on observing and imitating.

So our ranking system will evaluate every child separately based on some attributes as: The social picture and health of the child's family, the financial status of his parents, whether the child is supported by one or two parents, etc.

Objective:

To classify the child as "not recommended, recommended, very recommended, priority and special priority" using a machine learning model that can predict the category of each child based on his/her features.

Machine Learning algorithms used to solve the problem:

- **Logistic Regression :**

  Logistic regression is one of the most fundamental and widely used machine learning algorithms. It is not a regression algorithm but a probabilistic classification model[3].
  It is used to describe the data and to explain the relationship between the target variable and one or more other variables[4].

- **Decision Tree :**

  A decision tree is a largely used non-parametric effective machine learning modeling technique for regression and classification problems. To find solutions a decision tree makes sequential, hierarchical decision about the outcomes variable based on the predictor data[5].

- **Support Vector Machine :**

  It’s is an algorithm that outputs an optimal hyperplane that separates the plotted data points[6].

The dataset will be preprocessed before applying these previous machine learning models. The main two steps for preprocessing are using One-Hot-Encoding (creating dummy variables) technique because the dataset contains categorical variables so we need to convert them into binary variables, and searching for missing variables.
Some other steps like separating the target variable and splitting the dataset into training and testing sets were used.

Metrics

There are two main performance measures that are used for this dataset:
Testing accuracy and Prediction time. Testing accuracy will ensure that future observations will be classified correctly. The chosen model is the model with the highest test accuracy and with the least prediction time.

We will discuss each performance metric independently:

- Accuracy[7]

  Accuracy is the most widely used performance metric in binary classification problems. It's the number of correctly classified data points over the total number of observations. By using accuracy in training, we can ensure that our model uses the data in the best possible way (understands the complexity of the data and doesn't under fit), and when using accuracy in testing set we ensure that the model generalizes well (doesn't over fit). We will also use K-fold cross validation with the accuracy metric to ensure that our results are correct as much as possible.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

- Time

  We measured each model prediction time (time needed to predict new instances).

II. Analysis

Data Exploration

The dataset used for this project is "Nursery Data Set" from UCI Machine Learning Repository[2]. It is a textual file in '.csv' format. It was published in 1997 and it has 12960 rows and 8 columns, each column represent one feature of a child and his parents (a piece of information).
Let's take a look at first five rows in the dataset:

	parents	has_nurs	form	children	housing	finance	social	health	target
0	usual	proper	complete	1	convenient	convenient	nonprob	recommended	recommend
1	usual	proper	complete	1	convenient	convenient	nonprob	priority	priority
2	usual	proper	complete	1	convenient	convenient	nonprob	not_recom	not_recom
3	usual	proper	complete	1	convenient	convenient	slightly_prob	recommended	recommend
4	usual	proper	complete	1	convenient	convenient	slightly_prob	priority	priority

Figure 1. First Five Rows in the Dataset

Each row represents a data point: one child observation, each column specifies a specific aspect of a child (a piece of information). By looking at the first five rows we can see that columns contain information about occupation of parents and child's nursery, family structure and financial standing, and social and health picture of the family and others. All the columns contain categorical variables and they will be encoded to numerical values to be able to feed them to different machine learning models. Our target variable is the 'target' column (last column) which represents the state of each as (not recommended, recommended, very recommended, priority and special priority). Other columns are features that will be fed to the machine learning models.

A] Here is a detailed information about each column possible values:

Attribute Information:

parents: usual, pretentious, great_pret
has_nurs: proper, less_proper, improper, critical, very_crit
form: complete, completed, incomplete, foster
children: 1, 2, 3, more
housing: convenient, less_conv, critical
finance: convenient, inconv
social: non-prob, slightly_prob, problematic
health: recommended, priority, not_recom

Figure 2. Attribute Information

B] Examining each attribute value set is the best way to understand the data.
Another check for dataset information:

```
The 'Nursery Dataset' has 12960 rows and 9 columns.

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12960 entries, 0 to 12959
Data columns (total 9 columns):
parents     12960 non-null object
has_nurs    12960 non-null object
form        12960 non-null object
children    12960 non-null object
housing     12960 non-null object
finance     12960 non-null object
social      12960 non-null object
health      12960 non-null object
target      12960 non-null object
dtypes: object(9)
memory usage: 911.3+ KB
```

Figure 3. Dataset Information snippet from code results: No. of Rows, Columns, Column Names and their data types.

C] Checking unique values of the dataset:

```
[('parents', array(['usual', 'pretentious', 'great_pret'], dtype=object)),
 ('has_nurs',
  array(['proper', 'less_proper', 'improper', 'critical', 'very_crit'], dtype=object)),
 ('form',
  array(['complete', 'completed', 'incomplete', 'foster'], dtype=object)),
 ('children', array(['1', '2', '3', 'more'], dtype=object)),
 ('housing', array(['convenient', 'less_conv', 'critical'], dtype=object)),
 ('finance', array(['convenient', 'inconv'], dtype=object)),
 ('social', array(['nonprob', 'slightly_prob', 'problematic'], dtype=object)),
 ('health', array(['recommended', 'priority', 'not_recom'], dtype=object)),
 ('target',
  array(['recommend', 'priority', 'not_recom', 'very_recom', 'spec_prior'], dtype=object))]
```

Figure 4. Kind of values our features can take on

D] There is another method that was used to explore the data, which to find number of unique values in each column and finding the most repeated value, a sample of columns is shown in the following figure:

	parents	has_nurs	form	children	housing	finance	social	health	target
count	12960	12960	12960	12960	12960	12960	12960	12960	12960
unique	3	5	4	4	3	2	3	3	5
top	usual	critical	completed	more	critical	inconv	nonprob	recommended	not_recom
freq	4320	2592	3240	3240	4320	6480	4320	4320	4320

Figure 5. Dataset Description: No. of Rows and Unique Values, Top Repeated

We can find valuable insights using the above figure, for example: 'children' has 4 unique values, and the most repeated one is 'more' with frequency of 3240.

Exploratory Visualization

Data Distribution

A count plot was used to know how the data is distributed, indicate how many times does a value in a column occurs. This is the most suitable plot to start with as it is simple and easy to understand. It also gives us valuable information on how the values are distributed and the total count of each variable in each column.

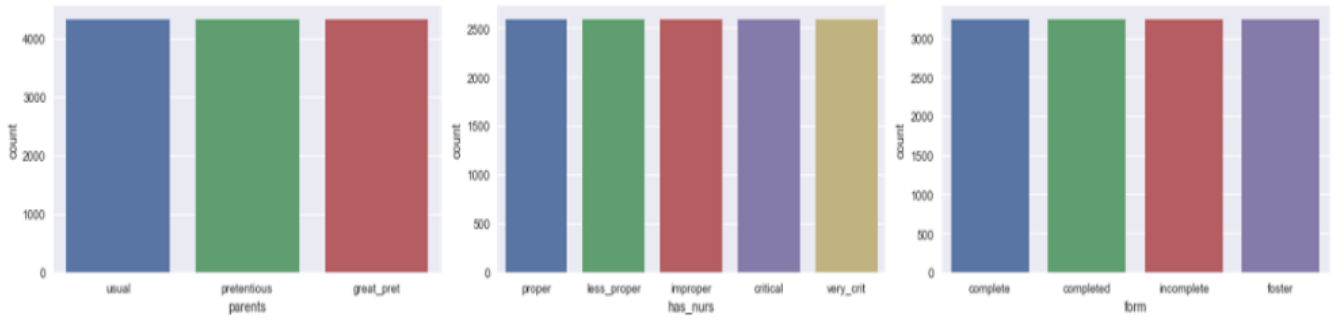

Figure 6. Sample of Data Distribution Plots

Bar Diagrams are plotted with column name at the bottom, x-axis holds column values and y-axis represents the total count of each value.
From the above figure we can see that each of our features is perfectly balanced with respect to the values they can take. The balance of classes are very important to avoid any bias during model building.

Plotting each feature against the target variable, 'target' Column

Now, it is time to find relationships between each variable and the 'target' column, this is done by plotting each column four times against the (not recommended, recommended, very recommended, priority and special priority) values of the target.

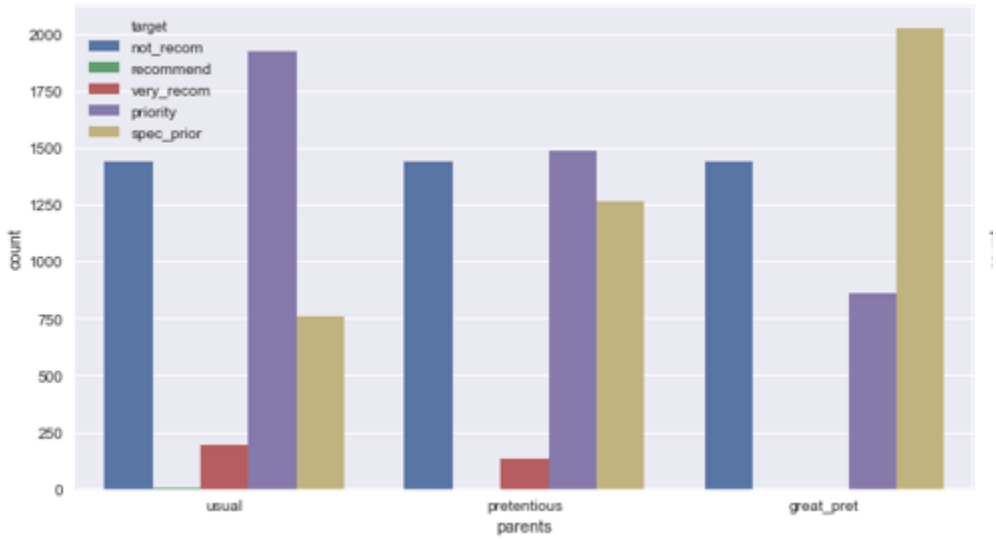

Figure 7. Parents' values against each target value

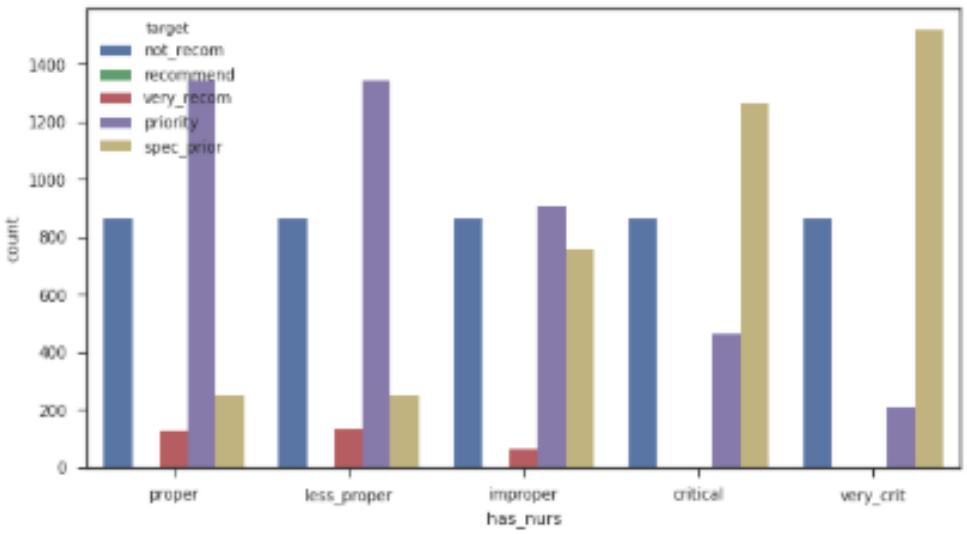

Figure 8. "has_nurs" values against each target value

These are some observation from this type of plots (including plots that are not in the above sample):

- There seems to be some correlation between the family's health status and the acceptance outcome.
- In the last plot, we clearly see that if the health status of the family does not recommend final acceptance, the final decision will be to not recommend the child.

To visualize this, and some other useful relations in the data, we will resort to use the correlation matrix between each values the features (and the target) can take. We'll display them using a heat-map to see what's going on:

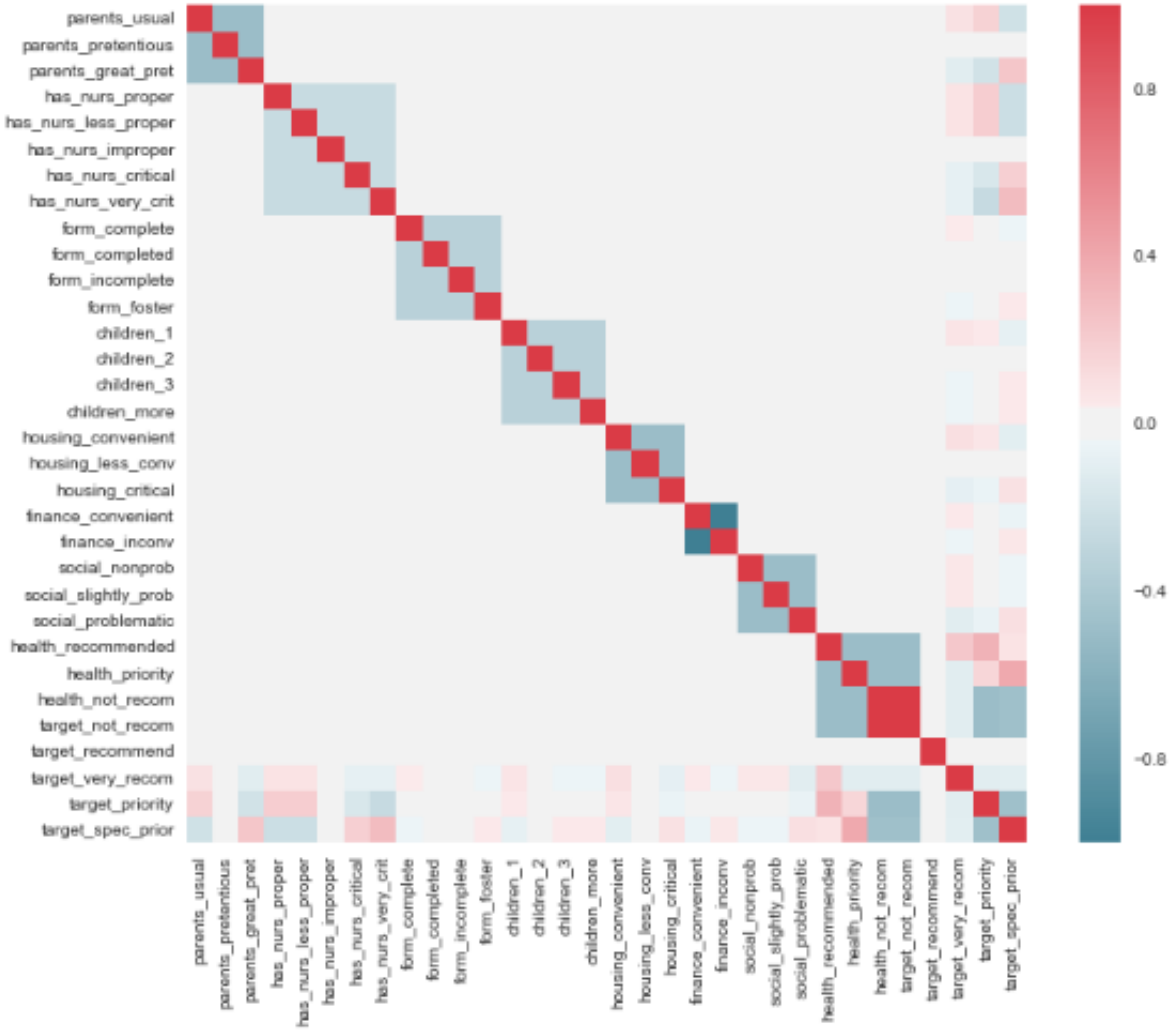

Figure 9. The correlation matrix between each value the features and the target can take.

We found that the correlation between target_not_recom and health_not_recom is 1.0, they are the same column.
At the same time, finance_convenient is perfectly negatively correlated
with finance_inconv which means that the two columns are complements.

Algorithms and Techniques

Nursery Application classification problem can be solved with supervised machine learning models, since all the data points are already labeled.

The four algorithms that I used to solve this problem are:

- Logistic Regression
- Decision Tree
- Support Vector Machine

1- Logistic Regression

Logistic Regression is a supervised classification model that is widely used as a base model which makes it a good candidate for our nursery application classification problem and that is how it works:

Logistic regression is a statistical technique used to predict probability of an event based on one or more independent variables. It means that, given certain factors (i.e. our training set), logistic regression will predict an outcome using maximum likelihood estimation to calculate the regression coefficients of the model and the model's features. It will be able to come up with z,
where (z = w0 + w1.x1 +w2:x2 + . . . + wn.xn) where (w0;w1;w2; :::;wn) are the regression coefficients and (x1; x2; x3; :::; xn) are our features.
After that the model calculates F(z) which is given by

$$F(z) = y(x)$$

$$y = \frac{1}{1 + e^{-x}}.$$

F(z) returns a probability value which can then be mapped to two or more discrete classes.

2- Decision Tree

Decision Tree Analysis is a general, predictive modelling. In general, decision trees are constructed via an algorithmic approach that identifies ways to split the data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks and we will be using it for classification in our case.
The decision rules are generally in form of if-then-else statements. The deeper the tree, the more complex the rules and fitter the model.
A decision tree is a tree-like graph with nodes representing the place where we pick an attribute and ask a question; edges represent the answers to the question; and the leaves represent the actual output or class label, and that is how it works:

Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example. Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new nodes.

I decided to go with Decision Tree because the data is categorical, and decision trees are in general a good choice when the data is categorical.

3- SVM

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. However, we will be using it for classification in our case. The idea of SVM is trying to find a hyperplane that best divides a dataset into classes. We can think of a hyperplane as a line that linearly separates and classifies a set of data. Choosing the right hyperplane that can separate the classes is based on the distance between the hyperplane and the nearest data point from either set which is known as the margin. The goal is to choose a hyperplane with the greatest possible margin between the hyperplane and any point within the training set, giving a greater chance of new data being classified correctly. Most of the time data don't have a clear boundary separating it, this is where it can get trick in order to classify a dataset that is not linearly separable. In simple terms the kernel moves away from a 2d view of the data to a 3d view by lifting data points into higher dimension making it possible to hang a horizontal plane between the two sets.
Support vector classifier was used for this problem because it is known to be robust and reliable.

Many machine learning algorithms cannot work with categorical data directly, so the categorical values were converted to numerical values by using one-hot- encoding method, this process converts categorical variables as binary vectors by creating a column for each possible value in dataset columns.
K-Fold cross validation technique is used to get more reliable scores.
In K-fold cross validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k = 1 subsamples are used as training data[8].

Benchmark

The chosen benchmark model for this project is a simple “Logistic Regression” classifier. The reason for choosing this specific algorithm is because it is simple and straightforward and doesn't require any hyper-parameters to specify beforehand, and it is so popular for classification problems that almost every machine learning engineer has used before. Also, 'Logistic Regression' is the most commonly reported data science method used at work for all industries except military and security where neural networks are used slightly more frequently according to Kaggle: The State of ML and Data Science 2017.[9]

III. Methodology

Data Preprocessing

Steps used to preprocess the data

1. Separating the target variable 'target' column into another variable to use it for splitting the dataset.

2. Checking for any missing data in the dataset, we get to know that there are no missing values in the dataset.

3. Using One-Hot-Encoding (creating dummy variables) technique to convert categorical variables into binary vector by creating a column for each possible value in dataset columns.

4. Splitting the dataset into training and testing sets with testing ratio of 20%.

The final preprocessed dataset has 19 columns (after One-Hot-Encoding) and 12960 rows.

Implementation

As mentioned in the previous sections, the dataset was loaded and analyzed to extract some useful information about it. Then, the dataset was cleaned and encoded as a part of

making it suitable for all models that we are going to use. Then the dataset was splitted into two parts: 80% training and 20% testing.
It is time to implement our machine learning models, this is done using three main steps:

- **Fitting the models**

  The ‘Logistic Regression’ was the first model to be trained on the dataset. After that, all the models were trained one by one ('Decision Tree', 'SVM').

- **Evaluating the models using training and testing sets**

  Here is the important part. Predicts the classes for training and testing sets by each model, and the prediction time was recorded (I used training sets predictions to determine if a model was under fitting). Accuracy performance metric was recorded for both training and testing sets for each model.

- **Comparing the results**

  As mentioned in the metrics section, our main evaluation metric will be the testing accuracy. After training the models and using them to predict new points and recording their performance metrics, a data frame whose keys are models and values are these metrics, I displayed all the results in an easy and convenient way for the reader to determine the best performing model. I used a table and a plot of results to take the final decision about the best classifier.

- **Enhancing the model with the highest accuracy and least prediction time**

  The chosen model is enhanced using hyper-parameter-tuning grid search technique.

Complications

The implementation process ran smoothly without any problems, the only thing that worth mentioning is that grid search took a lot of time relatively.

Results

	accuracy_test	accuracy_train	prediction_time
Decision Tree	0.995756	1.000000	0.000000
Logistic Regression	0.922840	0.914641	0.224124
SVM	0.968364	0.965856	1.844244

Figure 10. Models Evaluation Metrics

The above table shows that **Decision Tree** is the best model.

Decision Tree has an accuracy of 100% at training and 99.57% at testing but when using cross validation its accuracy is 99%. In terms of time, it was the fastest model to predict new data points.

SVM has 96.58% training accuracy and 96.83% testing accuracy but for cross validation its accuracy is 95.726%. It didn't take short time for prediction compared by the decision tree model, SVM has the longest prediction time.

Logistic Regression model scored 91.46% training accuracy and 92.28% testing accuracy with short prediction time.

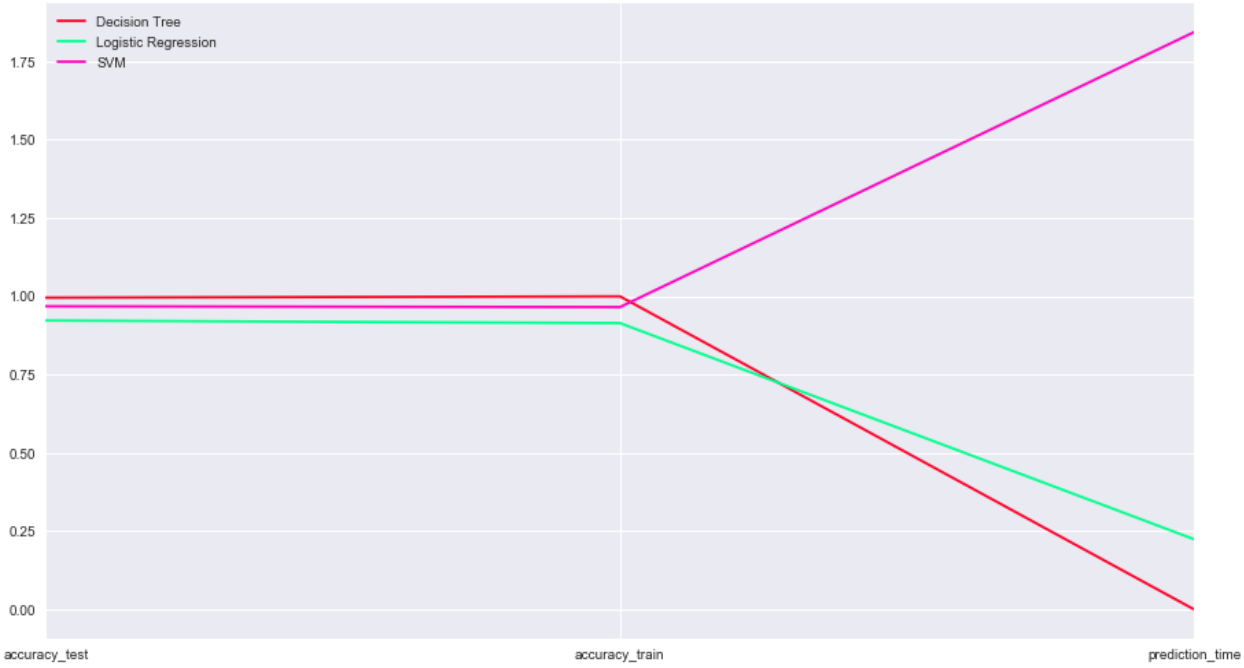

Figure 11. Model Evaluation Metrics Plot

Refinement

Hyper-parameter tuning was used to find out if there is better collection of parameters that may passed to the model and enhance the results, and to find the 'Feature importances'.

I have tweaked the following parameters:

- **Criterion**

  The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

- **Max_depth**

  The maximum depth of the tree. If none, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Decision Tree Results

Unoptimized Model Testing Accuracy: 0.9958
Unoptimized Model Cross Validation Score: 0.9905
Optimized Model Testing Accuracy: 0.9950
Optimized Model Cross Validation Score: 0.9917

	criterion	max_depth
best parameters values	entropy	None

Figure 12. Best Parameters after Hyper-Parameter Tuning

The optimized model has the same accuracy as the default one. Also, the optimized model has the same parameters after tuning. Even the results are the same, we can use the optimized model to find important features in our problem.

IV. Results

Model Evaluation and Validation

The final model has been chosen for this problem is 'Decision Tree Classifier' from sklearn library with its default parameters. I tried to hyper-parameter tune the model but the results was the same.

The model can be considered robust because its main performance rate (accuracy) was measured with two methods (train_test_split and cross validation)
Using train_test_split method with 'Decision Tree' output perfect scores with
100% accuracy, but this percentage cannot be trusted 100% because running the model with different random states will sometime change the percentage, and it depends heavily on how the data was splitted at first (especially when using random_state in train_test_split).
I used K-fold cross validation technique, which ensures that all the data points will be treated as
training or testing data point at some point of time, so by using this technique we trained the model several times and averaged the scores to be able to trust the final results and to consider the model as robust after knowing its actual performance and output.

Justification

My Benchmark model is a default logistic regression model that has an accuracy of 91.396% and prediction time (0.224 s)
The final model is 'Decision Tree' that has an accuracy of 99% which is higher than the benchmark model and has good performance rates and prediction time (0.00 s), and as we mentioned before, the accuracy and prediction time are our main focus. Our model has beaten the benchmark model in both metrics with high scores, hence it will be a good model for nursery application classification problem.

	accuracy_test	accuracy_train	prediction_time
Decision Tree	0.995756	1.000000	0.000000
Logistic Regression	0.922840	0.914641	0.224124

Figure 13. Comparison between the final model and benchmark model

V. Conclusion

Free-Form Visualization

Plotting Decision Tree feature importance

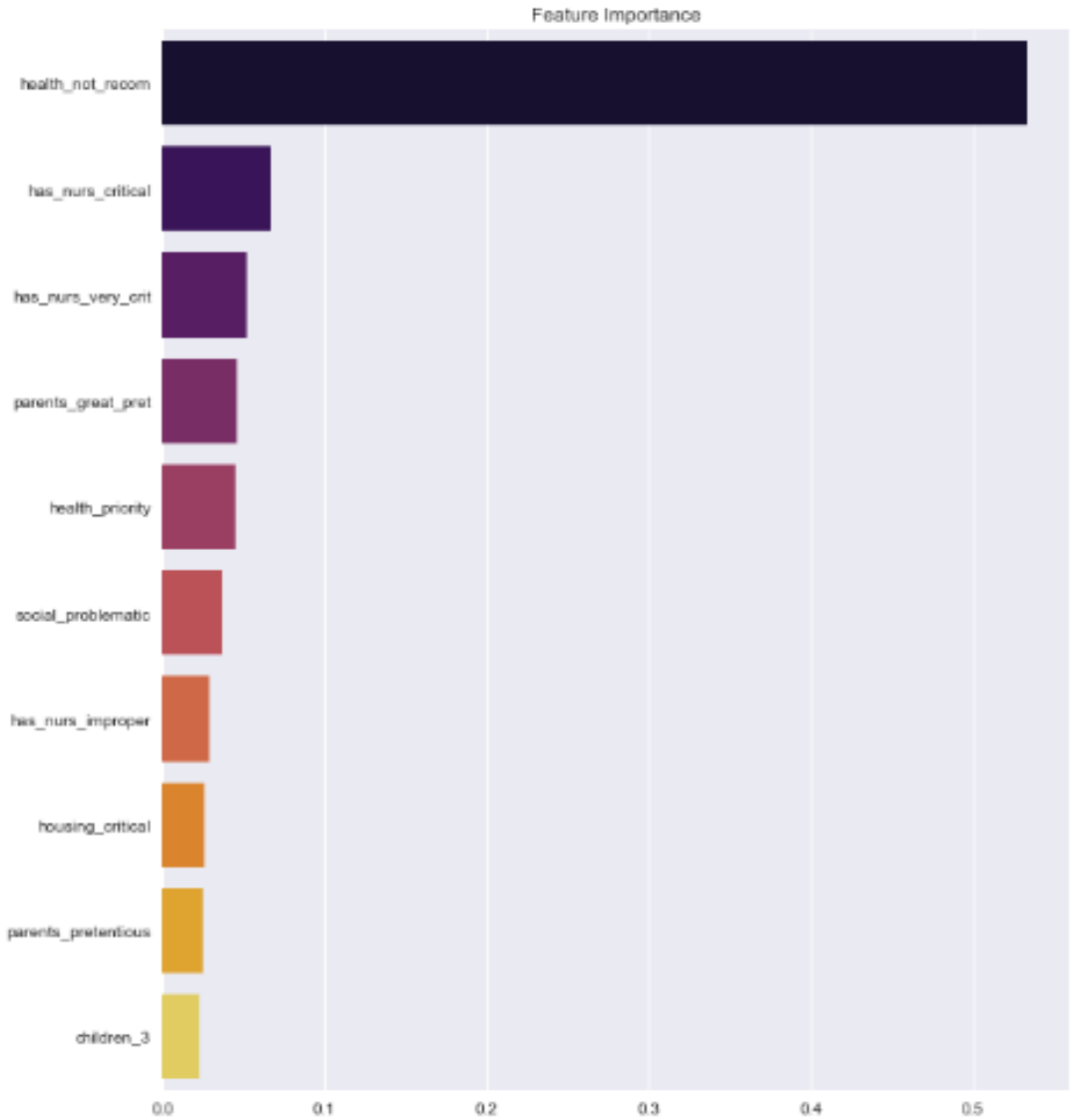

Figure 14. Feature Importance

Some values from 'health', 'has_nurs', 'parents', 'social', 'housing', 'children' where considered the most important factors in our classification. This is related by our observations during examining the relationship between each column and the target variable in the 'Data Analysis' section, it’s a good indicator that our model is reliable.

Reflection

Main Project Steps

1. Data Analysis: reading the dataset file and know more about the meta data.

2. Data Preprocessing: preparing dataset for machine learning models.

3. Split Dataset: splitting dataset into training and testing.

4. Build Benchmark Model: training 'Logistic Regression' model.

5. Building Different Machine Learning Models: training ‘Decision Tree’ and ‘SVM’ models
6. Evaluation: comparing different models results and choosing the best one among them.
7. Validation: enhancing the best model from the previous step.

Our problem was about classifying whether the child is "not recommended, recommended, very recommended, priority and special priority". It solves the problem of figuring out what prerequisites are needed for a child's application to be accepted by the nursery school. The dataset used for this problem was obtained from 'UCI Machine Learning Repository'. It is a textual data with 12960 rows and 8 columns. All the columns contain categorical values, the 'target' column was our target variable.

First: we started with data exploration to find the distribution of each column, then we plotted every column against the target variable to see if we can predict the behavior of the model.

Second: we preprocessed the data by using one-hot-encoding technique.

Third: we created the benchmark model (Logistic Regression), built two other models (Decision Tree and SVM) to make a comparison between all of them

Decision Tree was the best model because it scored accuracy of 100% at training and 99.57% at testing and to confirm that, we also used cross validation (99% cross validation accuracy).we implemented the hyper-parameter tuning technique to find the best parameters for our model and to find the important features in our data.

Finally: I found that count plot is the most suitable plot for our data.

Improvements

Improvements that will try do in future works:

- Doing feature reduction with PCA and possibly eliminating features that don't help in the classification process, and then comparing results with and without feature reduction.
- Using more values in hyper-parameter tuning to find better models.
- Using more complex models like neural networks and compare its results with ours.

References

[1] "Pros and cons of day nurseries," https://www.babycentre.co.uk/a537552/pros-and-cons-of-day-nurseries

[2] UCI Machine learning repository: "Nursery Data Set," https://archive.ics.uci.edu/ml/datasets/nursery?fbclid=IwAR0qzA8oKVP-4Sv9WRggMSwCUr8nqtj7AnJRMGGM4i5cLQnBUUg3loZxOpQ

[3] "Multivariate Multilabel Classification with Logistic Regression," https://acadgild.com/blog/logistic-regression-multiclass-classification

[4] "What is Logistic Regression?" https://www.statisticssolutions.com/what-is-logistic regression/

[5] "what is a Decision Tree?" https://towardsdatascience.com/what-is-a-decision-tree-22975f00f3e1

[6] "Chapter 2: SVM (Support Vector Machine) - Theory - Machine Learning 101 – Medium," https://medium.com/machine- learning- 101/chapter-2-svm-support-vector-machine-theory-f0812effc72.

[7] "Metrics to Evaluate your Machine Learning Algorithm," https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[8] "Cross-validation (statistics) – Wikipedia," https://en.wikipedia.org/wiki/Cross-validation_(statistics)#k-fold_cross-validation.

[9] "The State of ML and Data Science 2017 | Kaggle," https://www.kaggle.com/surveys/2017