|  |  |
| --- | --- |
| Internship Project Title | RIO-125: HR Salary Dashboard - Train the Dataset and Predict Salary |
| Name of the Company | TCS iON |
| Name of the Industry Mentor | Rushikesh |
| Name of the Institute | Indian Institution of Information Technology and Management of Kerala |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Start Date | End Date | Total Effort (hrs.) | Project Environment | Tools used |
| 12/03/2021 | 10/06/2021 | 136 | Jupyter notebook | Python 3 |

**TABLE OF CONTENT**

* Acknowledgements
* Objective
* Introduction / Description of Internship
* Internship Activities
* Approach / Methodology
* Assumptions
* Exceptions / Exclusions
* Charts, Table, Diagrams
* Algorithms
* Challenges & Opportunities
* Risk Vs Reward
* Reflections on the Internship
* Recommendations
* Outcome / Conclusion
* Enhancement Scope
* Link to code and executable file

**ACKNOWLEDGEMENTS**

First of all, I am expressing my sincere gratitude towards my industry mentor, Rushikesh and academic mentor, Dr. Manoj Kumar T.K for helping me throughout this project until now and providing me this wonderful platform to complete this project. I am also thankful for answering my queries at every phase of the project. You kept me going until now. I also want to thank all my friends who helped me with valuable suggestions during this project.

**OBJECTIVE**

The objective of this model is to make a salary prediction dashboard.

**INTRODUCTION**

My dataset contains age, workclass, education, education-num, marital-status, occupation, relationship, race, sex, capital-gain, capital-loss, hours-per-week, native-country attributes and a target attribute. During the first interim, 5 days of my project I cleaned and sanitized my dataset. On the next 10 days I went through several articles and videos regarding classification models and techniques. Then I developed my model using logistic regression, SVM and Random Forest. And I have made a comparison between output of these 3 models. At the end I have predicted the result using a user defined data tuple.

The model that I have developed is a HR -Salary dashboard, so the main objective of this model is to predict the salary of an employee who is newly admitted to certain department or position.

**INTERNSHIP ACTIVITIES**

* Attended the RIO – pre assessment test.
* Went through the helping materials available in our dashboard such as the Welcome Kit, Day wise plan, Project reference material etc.
* Watched the webinars and recorded lectures.
* Created a dataset that is suitable for this project.
* Cleaned and sanitized the dataset.
* Went through many articles and videos to learn about classification models and training techniques.
* Trained the dataset to predict the salary of a particular HR when they switch jobs.
* Made a comparison between 3 different classification techniques i.e. SVM, logistic regression, random forest.
* Wrote activity reports and project interim reports.

**METHODOLOGY**

So, the target attribute in my dataset contains only 2 classes. Hence, we need to predict our salary to be in one of these 2 classes. So, our model converges to a binary classification model. There are several methods to make a binary classification which include SVM, logistic regression, random forest etc. I have trained and tested my data using SVM, logistic regression and random forest and made a comparison among them.

**ASSUMPTIONS**

The dataset that I opted contained a value ‘?’ under some of the attributes, I assumed it to be used for the data which are unknown and hence I replaced all these values with ‘unknown’. Also, I have made an assumption that this data doesn’t contain any outliers in them. I believed this data to be true and not some manually created random dataset. I have removed 2 attributes containing capital loss and capital gain of the employees. I assume these attributes to not affect the target class since these events comes after the employee get paid.

**EXCLUSIONS**

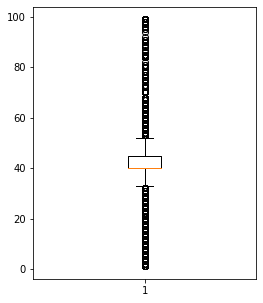
This project can not predict an exact salary of an employee who is switching job. This project would only classify his / her salary to be in one of the classes ‘>50K’ or ‘<=50K’. The dataset contains no data about the work experience of employees.

**CHARTS, TABLE, DIAGRAMS**

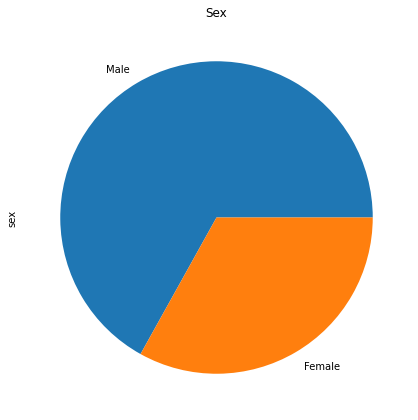
The following are charts and diagrams that I have created as part of the visualization.

**Bar graph plotting the native countries of the employees**

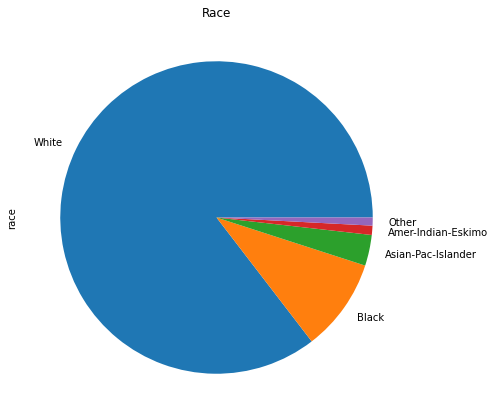
**Boxplot showing the no. of working hours (per week) of employees**



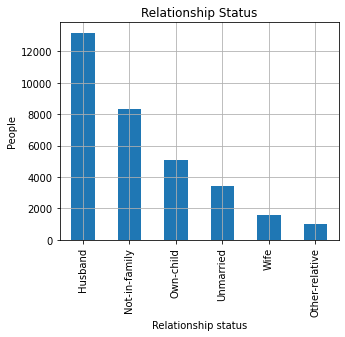
**Pie chart plotting gender distribution**



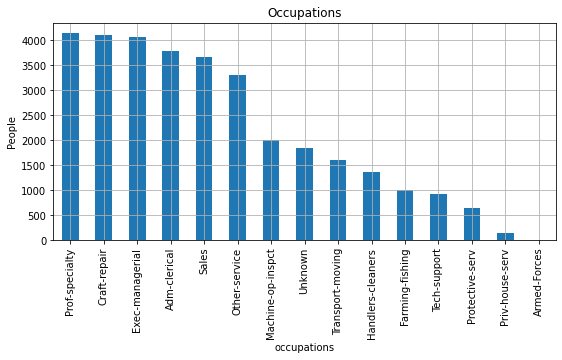
**Pie chart plotting race distribution**



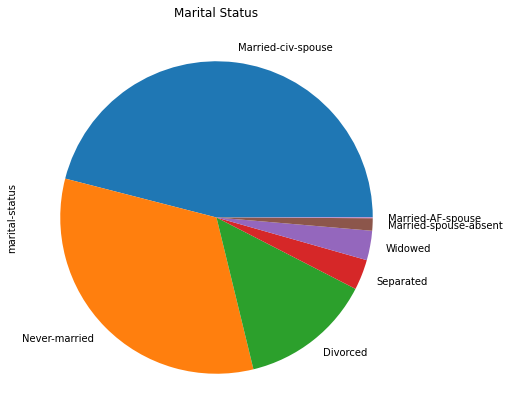
**Bar graph plotting the relationship status of the employees**



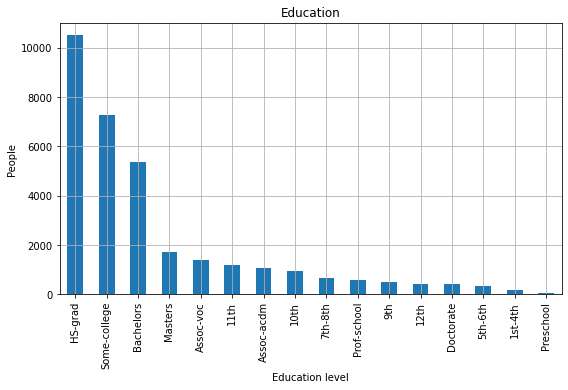
**Bar graph plotting the occupations of the employees**



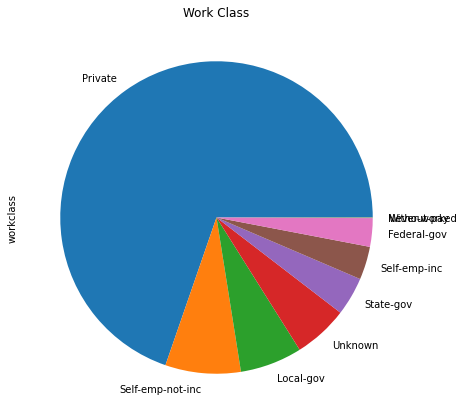
**Pie chart plotting the marital status of the employees**



**Bar graph plotting the education level of the employees**



**Pie chart plotting the work class distribution**



**The diagram showing comparison between 3 classification models that I have tried.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | Accuracy | f1 score |
| SVM | 0.81 | 0.82 | 0.82 | 0.81 |
| Logistic Regression | 0.74 | 0.77 | 0.77 | 0.74 |
| Random Forest | 0.81 | 0.81 | 0.81 | 0.81 |

**ALGORITHMS**

This is the algorithm for development of a classification model that I have used here

1. Start
2. Import necessary libraries.
3. Vectorize character values.
4. Normalize all the values.
5. Split training and testing data.
6. Did hyper parameter tuning with the corresponding classification method.
7. Trained the model.
8. Tested the model using test data.
9. End

**CHALLENGES & OPPORTUNITIES**

With this internship one of the main challenges that I had faced was on visualizing each attribute relations and developing the model. I had a rough idea on these classification techniques from my academic background. But these daily activities let me get a thorough idea on these techniques. So I had the opportunities to build a concrete base in myself about these models. And I also learnt from this internship to do hyperparameter tuning which I wasn’t aware of earlier.

**RISK VS REWARD**

The risk element in this project is that there are a lot of attributes considered in this project and the number of data tuples used to predict a class according to these many attributes are not enough. Since all these included features or attributes had some amount of role to play in predicting an individual’s salary, I had no other option but to keep them.

**REFLECTIONS ON THE INTERNSHIP**

I like the way they introduced a discussion room for all of us to connect with each other. Now, I am more adapted to systematic approach on working in a project. I got introduced to writing daily reports and interim reports. Also, I was happy to see how my industry mentors responded to my queries. It all went in a well-maintained structured routine.

**RECOMMENDATIONS**

Actually, this internship asked us to make a model to imitate a HR salary dashboard, but it just converged onto just a development of a python program. Rather it could have been a development of a working prototype, software or an application.

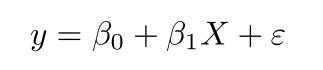
**OUTCOME / CONCLUSION**

During these 30 days of my project as mentioned in all the reports that I have submitted, I have gone through many articles and videos regarding Logistic regression, Linear regression, SVM, random forest etc. And I am noting down some of my learnings down here.

Also, I could enhance my programing skills through this project like visualization, sanitization of the dataset and on how to use the 3 classification models that I have used in this project.

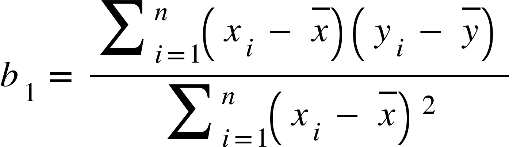
**Linear Regression**

Linear regression is perhaps one of the most well-known and well understood algorithms in statistics and machine learning. It is a kind of regression in which we try to fit a line onto a set of data points. It is one of the most common algorithms that’s used for predictive analysis. The base of the model is the relation between a dependent and independent variable basically represented as



* **y** is the predicted value of the dependent variable (**y**) for any given value of the independent variable (**x**).
* **B0** is the **intercept**, the predicted value of **y** when the **x** is 0.
* **B1** is the regression coefficient – how much we expect **y** to change as **x** increases.
* **x** is the independent variable (the variable we expect is influencing **y**).
* **e** is the **error** of the estimate, or how much variation there is in our estimate of the regression coefficient.

Linear regression finds the line of best fit line through your data by searching for the regression coefficient (B1) that minimizes the total error (e) of the model.

By differentiating the above formula, we can obtain an equation for beta1 and beta2 using which we can define the equation for error. Then we will try to define the model by minimizing the residual error.



[Gradient descent is an optimization algorithm that finds the values of parameters (coefficients) of a function (f) to minimize the cost function (cost)](https://machinelearningmastery.com/gradient-descent-for-machine-learning/).

**Logistic Regression**

 Logistic Regression is used when the dependent variable(target) is categorical.

The type of function used here is a sigmoid function.

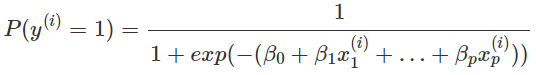
If ‘Z’ goes to infinity, Y(predicted) will become 1 and if ‘Z’ goes to negative infinity, Y(predicted) will become 0. So, the only outputs of a logistic regression model are ‘0’ and ‘1’.

logistic(η) =

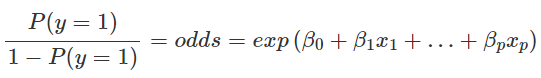
The step from linear regression to logistic regression is kind of straightforward. In the linear regression model, we have modelled the relationship between outcome and features with a linear equation:



For classification, we prefer probabilities between 0 and 1, so we wrap the right side of the equation into the logistic function. This forces the output to assume only values between 0 and 1.

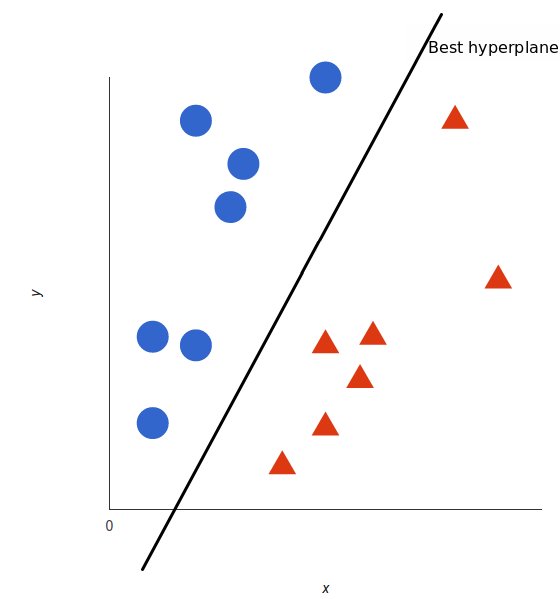


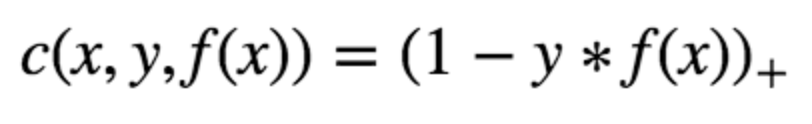
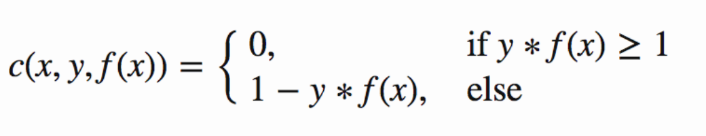
From the above equation, in the end we can define the odds ratio as

****

**SVM**

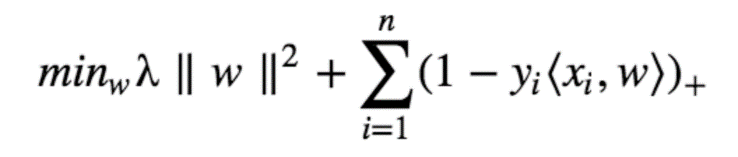
A support vector machine takes data points and outputs the hyperplane (which in two dimensions it’s simply a line) that best separates the tags. This line is the **decision boundary**: In the following example, anything that falls to one side of it we will classify as blue, and anything that falls to the other as red.

****

For SVM, the best hyper plane is the one that maximizes the margins from both tags. The loss function that helps maximize the margin is hinge loss.

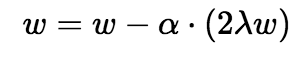
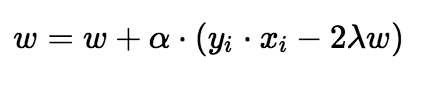
Hinge loss function (function on left can be represented as a function on the right)

Then, we have the loss function



Now that we have the loss function, we take partial derivatives with respect to the weights to find the gradients. Using the gradients, we can update our weights.

When there is no misclassification, i.e., our model correctly predicts the class of our data point, we only have to update the gradient from the regularization parameter. When there is a misclassification, i.e., our model makes a mistake on the prediction of the class of our data point, we include the loss along with the regularization parameter to perform gradient update.

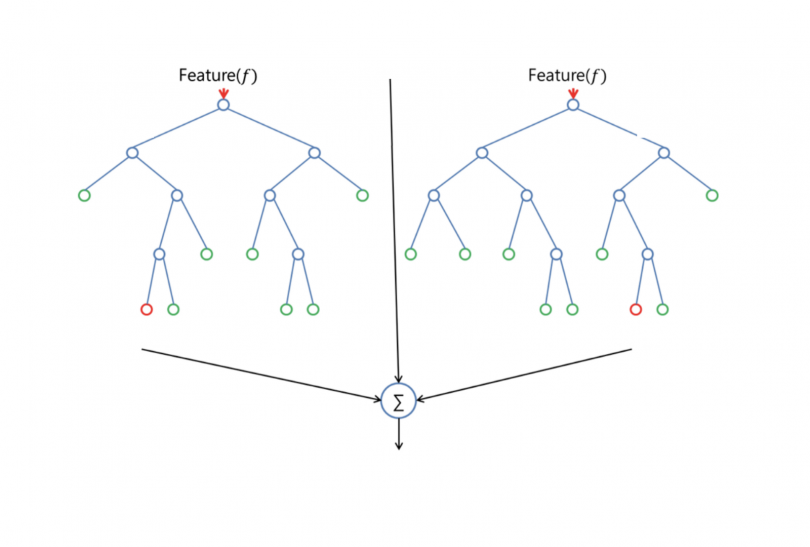


Gradient Update — Misclassification

Gradient Update — No misclassification

**Random Forest**

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.



Random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. Fortunately, there's no need to combine a decision tree with a bagging classifier because you can easily use the classifier-class of random forest. With random forest, we can also deal with regression tasks by using the algorithm's regressor.

Random forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model.

Therefore, in random forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. We can even make trees more random by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does).

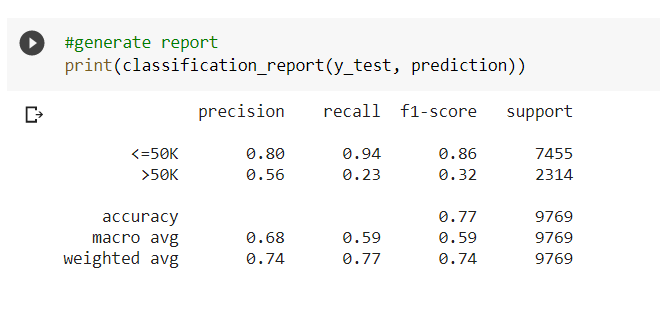
**Project Development**

Now at the end of this project, I have cleaned, sanitized, visualized and preprocessed the dataset. I have also trained and tested the model using logistic regression, SVM, random forest. Then I have created classification report. Then I have tuned the parameters of the model and chose the best ones. And again, I have tested and printed the classification report. Hence it was evident that the SVM model had more accuracy than the other 2. So I chose SVM classifier over others. At the end I have tested the model against a user defined data tuple also.

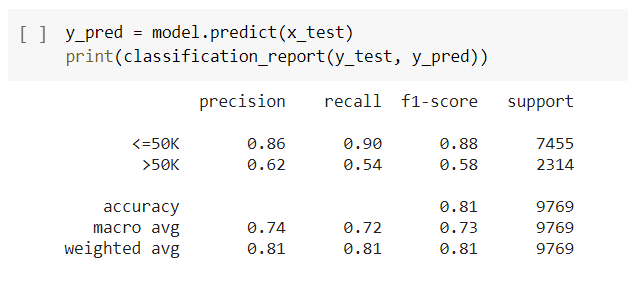
**Project Inference**

Here, in this project I tried it with 3 different models which are logistic regression, SVM and random forest.

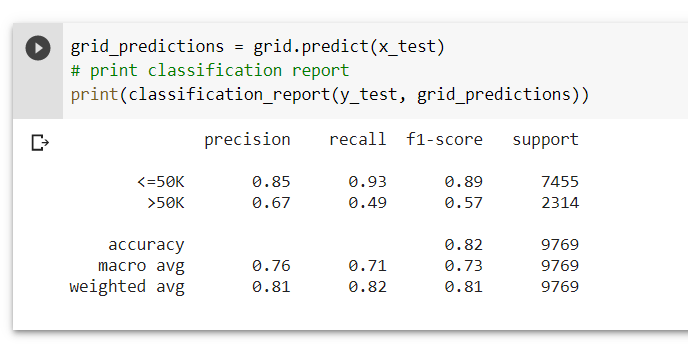
So, after hyper parameter tuning the classification report that I got from logistic regression model is

****

And the classification report for the random forest model is as follows.

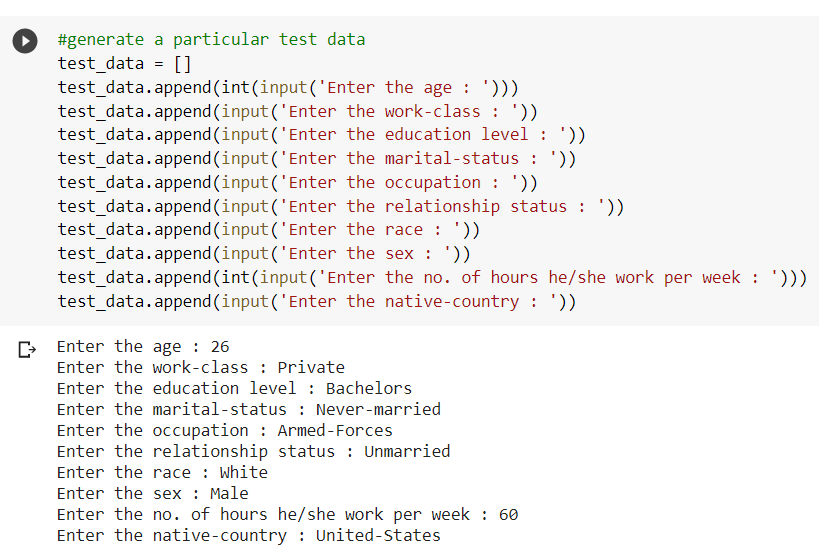
****

And the classification report for tuned SVM model is as follows.

****

And then I have compared these 3 classifiers according to the parameters f1 score, accuracy, precision and recall.

And its clear that the SVM model has more accuracy among the 3. Hence I chose the SVM model among these. Then I used the SVM model for testing on a particular case.

****

The output I got for this particular case was this.

****

To conclude, HR Salary Dashboard works like a very useful tool for predicting the salary of new employees. I think it can predict the salaries very well if we can include ‘years of experience’ attribute to the data. I think the model still lag behind in case of accuracy since the model could achieve an accuracy of only 82%. May be including more data might be a solution to the problem. Otherwise, I hope my project is up and ready for use if it’s converted to an application.

**ENHANCEMENT SCOPE**

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. Also, we have a very good client base as this can be useful in many of the organizations and companies. The project can also be converted to an application in the future.

The following are the future scope for the project.

* Addition of a ‘years of experience attribute’.
* Addition of a resume validator to this model.

**LINK TO CODE AND EXECUTABLE FILE**

Link to the colab file:

<https://colab.research.google.com/drive/1KOhRZ8AzNSDxyZ1Llr3yopWI1p9DGwEs?usp=sharing>

Link to the GitHub file:

<https://github.com/Rithik-Alias/Salary-prediction-dashboard>