Internship Project Title	TCS iON RIO-210: Salary Prediction Dashboard for HRs
Name of the Company	TCS iON
Name of the Industry Mentor	Dr.Himdweep Walia
Name of the Institute	Amity University Online

Start Date	End Date	Total Effort (hrs.)	Project Environment	Tools used
02/12/2024	29/01/2025	170	Jupyter notebook	Python 3

TABLE OF CONTENT

ACKNOWLEDGMENTS

I, Archita Debnath, a student pursuing a Master of Computer Applications with a specialization in Machine Learning, hereby declare that I have prepared My Internship Project titled, **“TCS iON RIO-210: Salary Prediction Dashboard for HRs,”** under the guidance of Dr.Himdweep Walia & Mr. Avinash Singh.

First and foremost, I sincerely express my gratitude to my industry mentor, Dr.Himdweep Walia and academic mentor, Mr. Avinash Singh, for their valuable support, guidance, and encouragement throughout this project. Their expertise and insights were crucial in helping me achieve the objectives of this work. I am deeply thankful for their availability and for patiently addressing my queries at every phase of the project.

OBJECTIVE

The objective of this project is to train machine learning models to accurately predict salaries and to develop a user-friendly, efficient salary prediction dashboard specifically designed for HR professionals.

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 milestone, 7 days of my project, I collected and cleaned the dataset, ensuring it was properly cleaned, sanitized and ready for training. The dataset is now prepared for use in salary prediction. Over the next 15 days, I conducted exploratory data analysis by visualizing the relationship between various attributes and the salary of HR professionals. Subsequently, I trained the dataset using machine learning models, including Logistic Regression, Random Forest, and Support Vector Machine (SVM). Additionally, I performed hyperparameter tuning to optimize model performance. After that, I have generated the classification report. Finally, I have predicted the result using a user-defined data tuple.

After that, I created an interactive dashboard on Power BI to better understand the project.

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.
- Created a Power BI Dashboard.

APPROACH / METHODOLOGY

Project methodology refers to the structured approach and systematic processes undertaken to achieve the objectives of a project. It typically involves defining the problem, collecting and preprocessing data, performing exploratory data analysis (EDA), developing and optimizing models, and deploying the solution. In this project, the methodology included sourcing a dataset, cleansing and analyzing the data, applying machine learning algorithms such as Logistic Regression, Random Forest, and SVM, conducting hyperparameter tuning for optimization, validating model performance with metrics, and integrating the final model into a dashboard for real-time salary prediction by HR professionals. This streamlined approach ensures efficient problem-solving and actionable outcomes.

ASSUMPTIONS

The dataset that I opted for 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 assumed 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 do not affect the target class since these events come after the employee get paid.

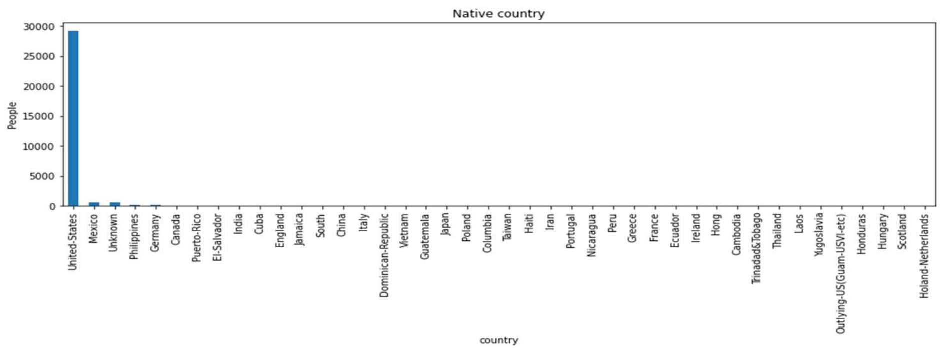
EXCLUSIONS

This project cannot predict the 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.

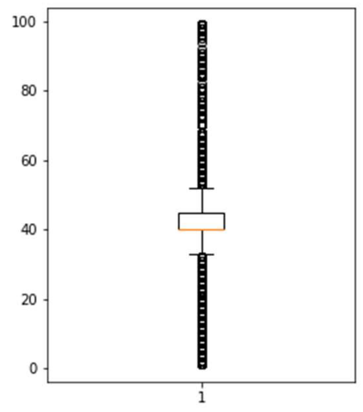
CHARTS, TABLE, DIAGRAMS

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

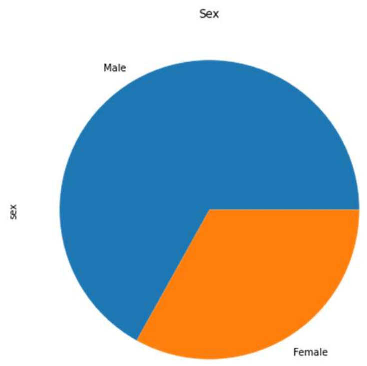
Bar graph plotting the native countries of the employees



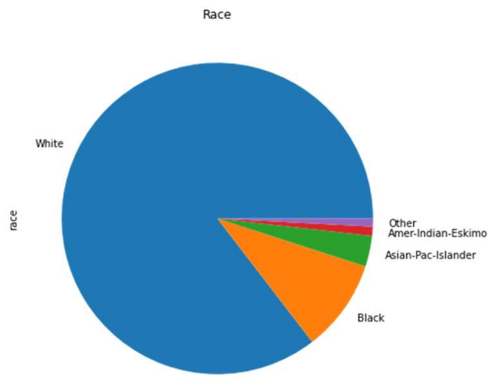
Boxplot showing the no. of working hours (per week) of employees



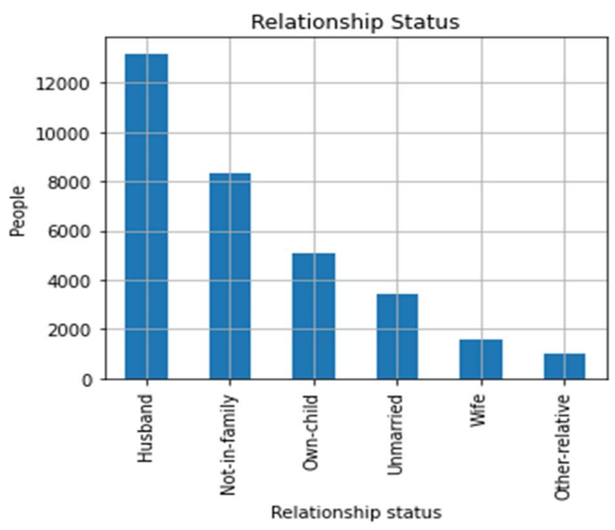
Pie chart plotting gender distribution



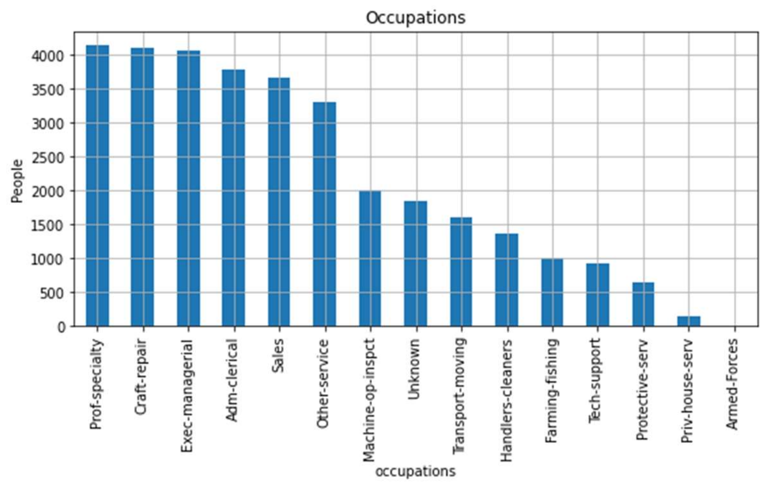
Pie chart plotting race distribution



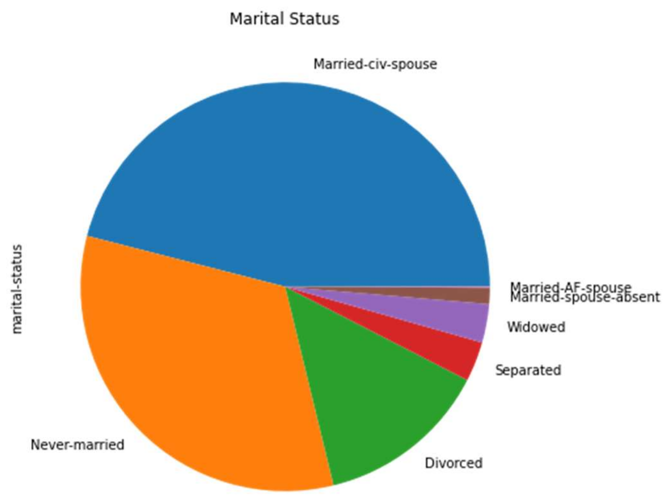
Bar graph plotting the relationship status of the employees



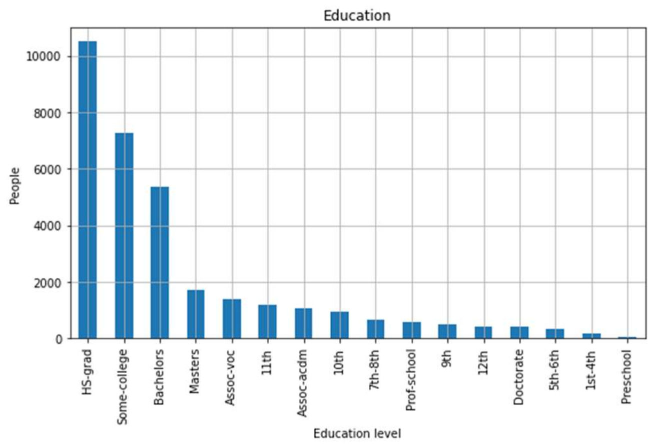
Bar graph plotting the occupations of the employees



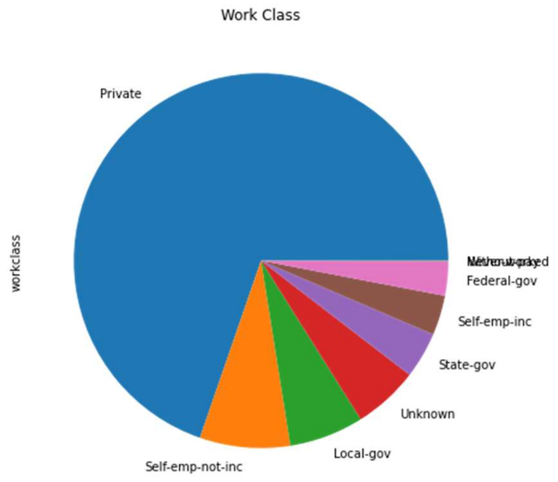
Pie chart plotting the marital status of the employees



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.

	Accuracy
SVM	0.82
Logistic Regression	0.77
Random Forest	0.82

ALGORITHMS

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

1. Import data & import necessary libraries.
2. Vectorize character values.
3. Normalize all the values.
4. Split training and testing data.
5. Did hyperparameter tuning with the corresponding classification method.
6. Trained the model.
7. Tested the model using test data.

CHALLENGES & OPPORTUNITIES

During this internship, one of the main challenges I faced was visualizing the relationships between attributes and developing the model. Although I had a basic understanding of classification techniques from my academic background, working on this project gave me a deeper and more practical understanding of these methods. It provided me with the opportunity to build a strong foundation in these techniques. Additionally, I learned about hyperparameter tuning, a concept I was previously unfamiliar with, which has greatly enhanced my skills.

RISK VS REWARD

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

REFLECTIONS ON THE INTERNSHIP

I appreciated the introduction of a discussion room that allowed all of us to connect and collaborate effectively. Through this internship, I have become more accustomed to taking a systematic approach to working on projects. I also gained experience in writing daily reports and interim reports, which helped me stay organized and track progress. Additionally, I was happy to see how my industry mentors responded to my queries. It all went in a well-maintained structured routine.

RECOMMENDATIONS

The internship tasked us with creating a model to simulate an HR salary dashboard. However, the project primarily focused on developing a Python program. It would have been more impactful if the scope had been extended to developing a functional prototype, software, or application, which could better align with the intended objectives.

OUTCOME / CONCLUSION

During the 45 days of my project, as documented in the reports I have submitted, I explored various articles and videos to understand concepts such as Logistic Regression, Linear Regression, SVM, and Random Forest. I have noted some of my key learnings below.

Additionally, this project helped me improve my programming skills in areas such as data visualization, dataset sanitization, and the practical application of the three classification models I used in this work.

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

$$y = \beta_0 + \beta_1 X + \varepsilon$$

- **y** is the predicted value of the dependent variable (**y**) for any given value of the independent variable (**x**).
- **B$_0$** is the **intercept**, the predicted value of **y** when the **x** is 0.
- **B$_1$** 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 (B_1) that minimizes the total error (e) of the model.

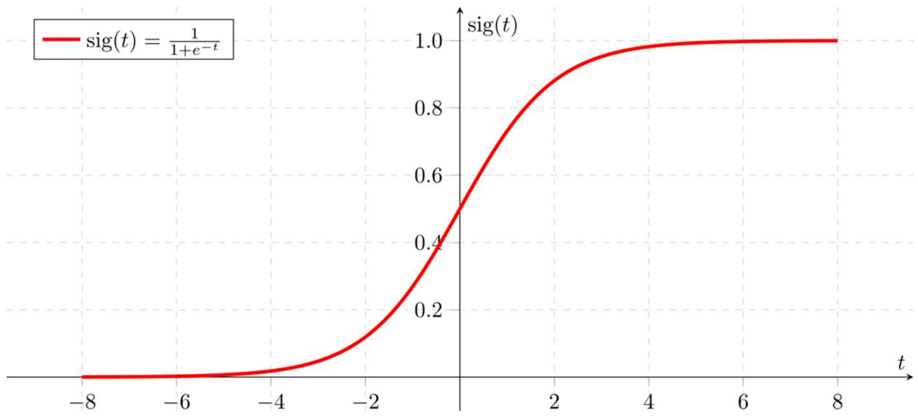
By differentiating the above formula, we can obtain an equation for beta1 and beta2 using which we can define the equation for error. Then we will try to define the model by minimizing the residual error.

$$b_0 = \bar{y} - b_1\bar{x} \qquad b_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

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’.

$$\text{logistic}(\eta) = \frac{1}{1+\exp(-\eta)}$$

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:

$$\hat{y}^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}$$

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.

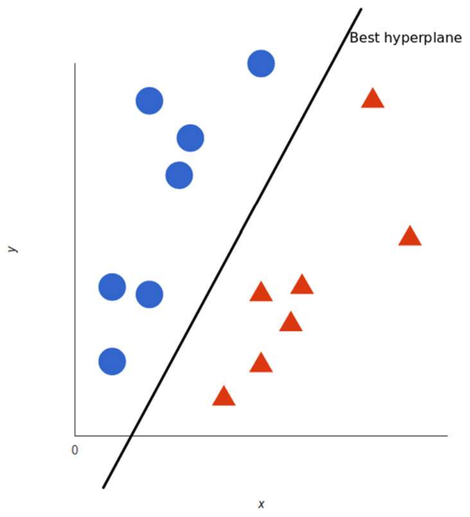
$$P(y^{(i)}=1)=\frac{1}{1+exp(-(\beta_0+\beta_1 x_1^{(i)}+\ldots+\beta_p x_p^{(i)}))}$$

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

$$\frac{P(y=1)}{1-P(y=1)}=odds=exp\left(\beta_0+\beta_1 x_1+\ldots+\beta_p x_p\right)$$

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 hyperplane is the one that maximizes the margins from both tags. The loss function that helps maximize the margin is hinge loss.

$$c(x,y,f(x))=\begin{cases}0, & \text{if } y*f(x)\geq 1\\ 1-y*f(x), & \text{else}\end{cases} \quad c(x,y,f(x))=(1-y*f(x))_+$$

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

Then, we have the loss function

$$min_w \lambda \parallel w \parallel^2 + \sum_{i=1}^{n}(1 - y_i \langle x_i, w \rangle)_+$$

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.

$$w = w - \alpha \cdot (2\lambda w)$$

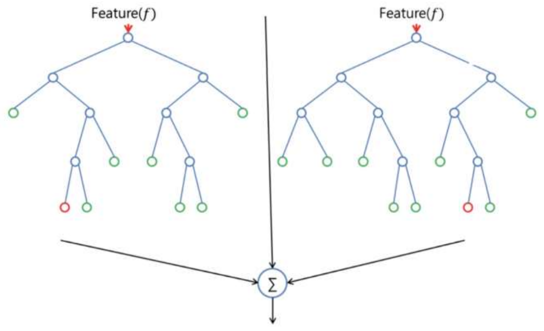
Gradient Update — No misclassification

$$w = w + \alpha \cdot (y_i \cdot x_i - 2\lambda w)$$

Gradient Update — 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).

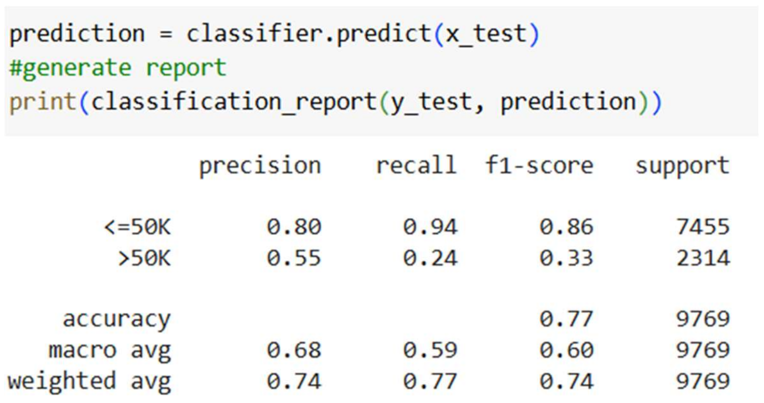
Project Development

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 and Random Forest. Then I have created classification report. Then I 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 and Random Forest models had more accuracy than the Logistic Regression model. So, here, 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 hyperparameter tuning the classification report that I got from the logistic regression model is



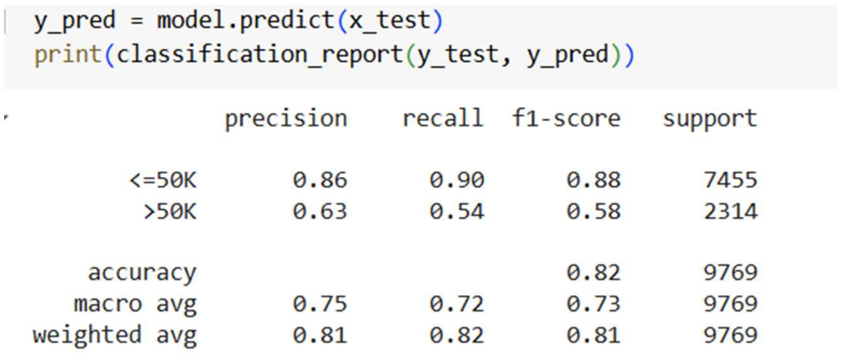
```
prediction = classifier.predict(x_test)
#generate report
print(classification_report(y_test, prediction))
```

```
              precision    recall  f1-score   support

        <=50K       0.80      0.94      0.86      7455
         >50K       0.55      0.24      0.33      2314

     accuracy                           0.77      9769
    macro avg       0.68      0.59      0.60      9769
 weighted avg       0.74      0.77      0.74      9769
```

And the classification report for the Random Forest model is as follows.



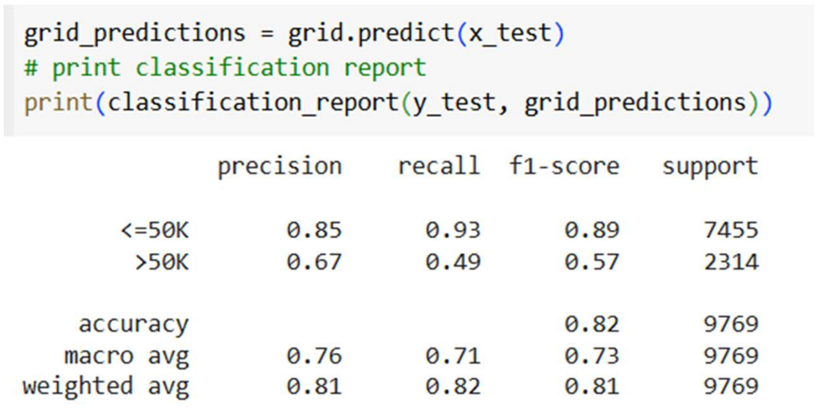
```
y_pred = model.predict(x_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

       <=50K       0.86      0.90      0.88      7455
        >50K       0.63      0.54      0.58      2314

    accuracy                           0.82      9769
   macro avg       0.75      0.72      0.73      9769
weighted avg       0.81      0.82      0.81      9769
```

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



```
grid_predictions = grid.predict(x_test)
# print classification report
print(classification_report(y_test, grid_predictions))
```

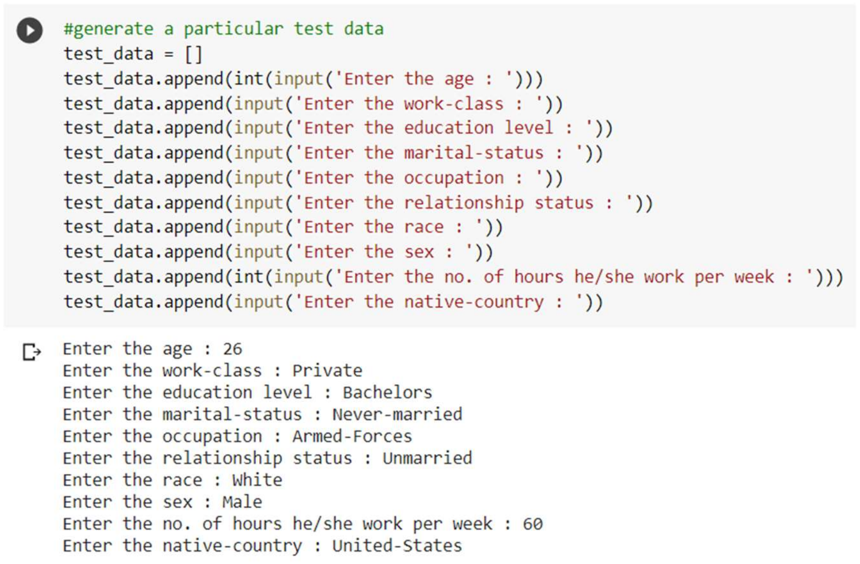
```
              precision    recall  f1-score   support

       <=50K       0.85      0.93      0.89      7455
        >50K       0.67      0.49      0.57      2314

    accuracy                           0.82      9769
   macro avg       0.76      0.71      0.73      9769
weighted avg       0.81      0.82      0.81      9769
```

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

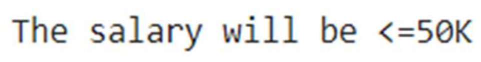
Hence, I chose the SVM model among these. Then I used the SVM model for testing on a particular case.



```
#generate a particular test data
test_data = []
test_data.append(int(input('Enter the age : ')))
test_data.append(input('Enter the work-class : '))
test_data.append(input('Enter the education level : '))
test_data.append(input('Enter the marital-status : '))
test_data.append(input('Enter the occupation : '))
test_data.append(input('Enter the relationship status : '))
test_data.append(input('Enter the race : '))
test_data.append(input('Enter the sex : '))
test_data.append(int(input('Enter the no. of hours he/she work per week : ')))
test_data.append(input('Enter the native-country : '))
```

```
Enter the age : 26
Enter the work-class : Private
Enter the education level : Bachelors
Enter the marital-status : Never-married
Enter the occupation : Armed-Forces
Enter the relationship status : Unmarried
Enter the race : White
Enter the sex : Male
Enter the no. of hours he/she work per week : 60
Enter the native-country : United-States
```

The output I got for this particular case was this.



```
The salary will be <=50K
```

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 lags behind in case of accuracy since the model could achieve an accuracy of only 82%. 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 lot of scope for future development and applications. It can be implemented on an intranet and easily updated as new requirements arise, due to its flexible design. With a strong client base, this project holds value for various organizations and companies. Moreover, it can be further developed into a standalone application.

The future scope of the project includes the following:

- Adding the "years of experience" attribute would further improve predictions.
- The integration of a resume validator feature in the model.

Link to code and executable file

Link to the colab file:

https://colab.research.google.com/drive/1b3j-E2XSyrCbPhyy6aM3A4YlXxUowgoM

Link to the GitHub file:

https://github.com/Archita05D/TCS-ION-RIO-210-Salary-Prediction-Dashboard