HLD Document

Campus Placement

Name: Abhishek Rajesh Saste

**Content**

[**Abstract** 3](#_Toc120548669)

[**1.** **Introduction** 3](#_Toc120548670)

[**2.** **General Description** 4](#_Toc120548671)

[**2.1** **Problem Statement** 4](#_Toc120548672)

[**2.2** **Proposed solution** 4](#_Toc120548673)

[**2.3** **Further improvements** 4](#_Toc120548674)

[**2.4** **Data requirement** 5](#_Toc120548675)

[**2.5** **Tools used** 5](#_Toc120548676)

[**3.** **Design Details** 5](#_Toc120548677)

[**3.1** **Process flow** 5](#_Toc120548678)

[**3.2** **Machine learning algorithms** 7](#_Toc120548679)

[**3.3** **Model selection** 7](#_Toc120548680)

[**4.** **Model Training and Evaluation** 9](#_Toc120548681)

[**5.** **Deployment Process** 10](#_Toc120548682)

[**6.** **Conclusion** 12](#_Toc120548683)

# **List of Figures**

[**Figure 1: System Flow Chart** 7](#_Toc120551142)

[**Figure 2: Scores for different models** 9](#_Toc120551143)

[**Figure 3: Parameter tuning of the best three models** 9](#_Toc120551144)

[**Figure 4: Cross-Validation scores for Support vector classifier** 10](#_Toc120551145)

[**Figure 5: Training accuracy of the Support Vector Classifier** 10](#_Toc120551146)

[**Figure 6: Testing accuracy of Support Vector Classifier** 10](#_Toc120551147)

[**Figure 7: Confusion matrix** 11](#_Toc120551148)

[**Figure 8: Steps to launch an instance in AWS** 12](#_Toc120551149)

# **Abstract**

The placement process is a very rugged process for recruiters as well as for students, on one side where recruiters sort the students based on their academic scores and experience it becomes important for students also to know before applying to any company what are their chances of getting placed. This problem can be overcome by providing students with a web app where, based on their academic scores and experiences they can get feedback on getting placed or not. In this project, the dataset containing different parameters was trained on SVC (Support Vector Classifier) after comparing the results with other machine-learning algorithms to get the best-fit algorithm with the highest accuracy. The models' accuracy was achieved using a grid search cross-validation method where the accuracy of SVC was also increased by tuning the model's parameters. The model was then used on test data which gave a testing accuracy of 79.06% and a training accuracy of 91.27%. A website was developed for the user interface and was deployed on AWS on the EC2 instance.

# **Introduction**

Campus placement is very important from the student’s perspective as it is one of the biggest steps in their career to get into their dream company. Machine learning is used in human resources in many accepts, to define recruitment patterns, and to help students as well as recruiters to know their outcomes in placements. This can be achieved by feeding different selection criteria to a machine-learning model, helping students know where they stand when applying for a specific company. The same concept is used in this project where, a website is created for students to know whether they will be placed based on different criteria, for example, 10th class, HSC, Degree percentage, work experience, etc. In this project, different machine algorithms were trained. The model with the highest accuracy was used for the prediction and also for the user interface. To achieve this, the data were explored, analyzed, cleaned, and trained on different ML algorithms. The highest accuracy-giving model was determined by the Grid search cross-validation technique. An attempt was also made for increasing the accuracy by tuning the parameters. The test data was then visualized on the confusion matrix to visualize the accuracy better. SVC showed the highest accuracy of all the algorithms which includes Random forest, Decision tree classifier, Support vector classifier and the model with the highest accuracy (SVC) was then deployed on a cloud platform.

# General Description

## Problem Statement

The reputation and yearly admissions of an institution invariably depend on the placements it provides its students with. That is why all the institutions, arduously, strive to strengthen their placement department so as to improve their institution as a whole. Any assistance in this area will positively impact an institution’s ability to place its students. This will always be helpful to both the students, as well as the institution. The main goal is to predict whether the student will be recruited for campus placements or not based on the available factors in the dataset.

## Proposed solution

To solve this problem, the data can be trained on one predicting factor and a machine learning model can be developed for the prediction. To achieve this, different machine learning algorithms can be trained. The algorithm with the highest accuracy can be used to predict the outcome of the student’s status of being placed in campus recruitment. Finally the user interface can be developed and deployed on an EC2 instance of AWS.

## Further improvements

The User interface can be made more interactive by adding various features such as:

* Prediction of the salary based on the student’s academic performance.
* Chances in the percentage of getting placed in certain companies.
* Provide a list of companies that are suitable for the candidates according to their relevant knowledge and experience.

## Data requirement

The prediction factor depends on different criteria present in the dataset to solve the problem statement. This dataset has more different classes in some rows in the string, so it needs to be converted into numbers so that the algorithm can understand. For this, label encoding is used wherein the alphabetical data is converted into numbers. Null values present in the dataset were made 0. The final dataset for training the model was finalized by dropping some criteria which do not made more effect on the prediction factor as per the problem statement.

## Tools used

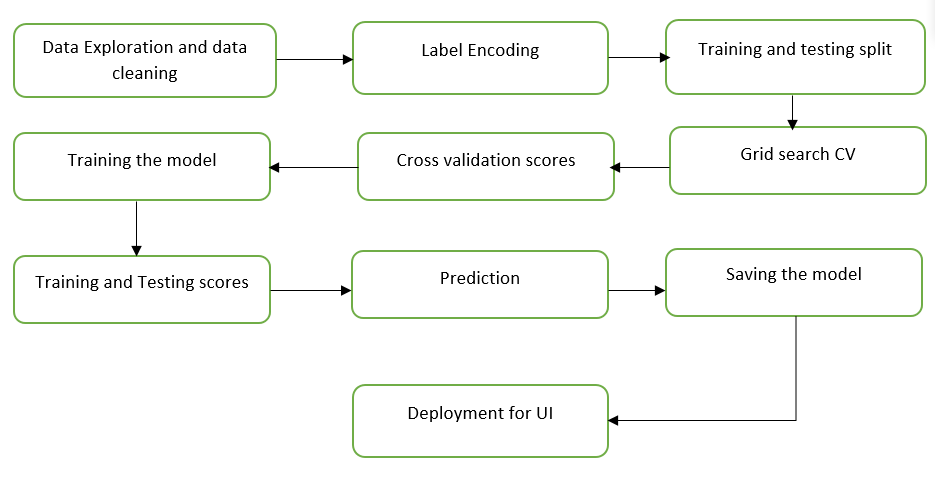
The following tools were used for the completion of the problem statement

1. Python
2. Numpy and Pandas for data cleaning
3. Matplotlib for data visualization
4. Sklearn for model building
5. Jupyter notebook, visual studio code as IDE
6. Python flask for HTTP server HTML/CSS for UI
7. AWS EC2 instance for deployment.

# Design Details

## Process flow

1. Fig 1 shows a flow chart of the system where the process begins with the data exploration and cleaning, here the data was observed and different accepts of the data were explored.
2. Further the data containing all the null values were made equal to 0.
3. Some features in the dataset were having string values which were then converted into integers using the label encoding method.
4. The final dataset was prepared by dropping the features which had less impact on the prediction factor.



**Figure 1: System Flow Chart**

1. The ratio for the training and testing split was chosen to be an 80:20 ratio which is the most common ratio used for training and testing the machine learning models.
2. Grid search cross-validation technique was used to find the best model from the different used algorithms.
3. Cross-validation scores were then observed for the highest training accuracy achiever model.
4. The model with the highest accuracy scores obtained from the grid search cross-validation technique was then trained and training, as well as testing accuracies, were obtained.
5. The predictions were then observed using the confusion matrix which showed the number of correct and incorrect predictions made by the trained model.
6. The model was then saved and used for the deployment of the users.

## Machine learning algorithms

The following dataset was trained on different models to find out the best model which can be used for the prediction. This totally depended on the accuracy obtained from the models.

Following were the models used for the training:

1. Random forest
2. Decision tree classifier
3. Support vector classifier

## Model selection

The best model from the listed algorithms was decided by the Grid search CV method. All the models with their different parameters were used in the process.

* Following were the parameters of the different algorithms:

1. Random forest:

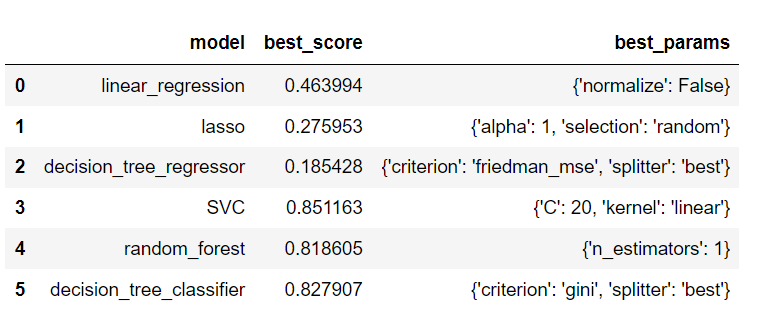
Parameters: n\_estimators: [1, 5]

1. Decision tree classifier:

Parameters: criterion: [gini, entropy]; splitter: [best]

1. Support vector classifier:

Parameters: C: [1, 20]; kernel: [linear]



**Figure 2: Scores for different models**

* Parameter tuning for the models with the highest accuracies:

1. Random forest:

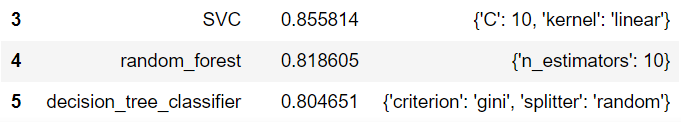
Parameters: n\_estimators: [1, 5, 10]

1. Decision tree classifier:

Parameters: criterion: [gini, entropy, log\_loss]; splitter: [best, random]

1. Support vector classifier:

Parameters: C: [1, 10, 20]; kernel: [rbf, linear]



**Figure 3: Parameter tuning of the three models**

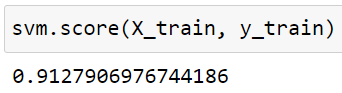
* The Cross-Validation Scores for Support Vector Classifier with ShuffleSplit as, n\_splits = 5, test\_size = 0.20, random\_state = 10 is shown in fig



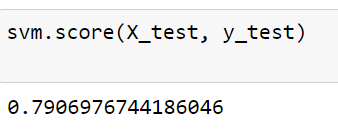
**Figure 4: Cross-Validation scores for Support vector classifier**

# Model Training and Evaluation

SVC showed the highest accuracy from all the models and hence it was used for the training and testing. The dataset was divided into 80:20 ratio for the training and testing requirements respectively. Fig shows the training accuracy of SVC, whereas fig shows the testing accuracy of SVC

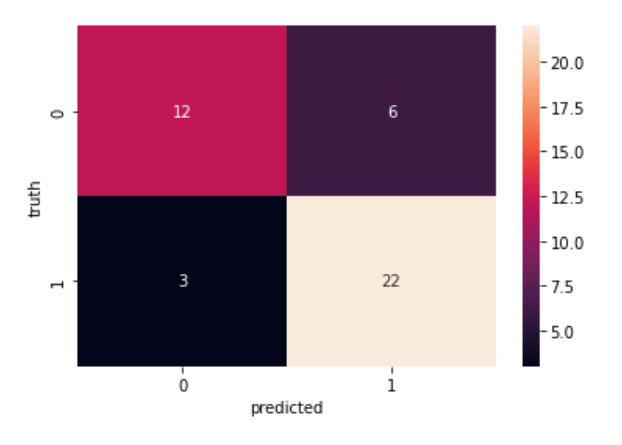


**Figure 5: Training accuracy of the Support Vector Classifier**



**Figure 6: Testing accuracy of Support Vector Classifier**

For better visualization confusion matrix was used to get the prediction numbers on the testing dataset which is shown in the figure



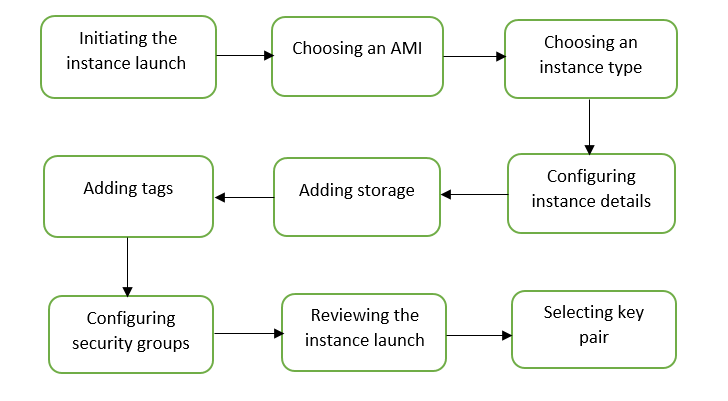
**Figure 7: Confusion matrix**

As seen in the figure, out of 43 inputs in total 34 were predicted correct whereas, 9 inputs were detected wrong.

# Deployment Process

The deployment of the UI is being made on Amazon Web Service’s EC2 instances. To achieve this the user interface was developed with the help of Python FLASK as an HTTP server using HTML and CSS. The deployment process for on the AWS is shown in figure. Following are the steps for launching an EC2 instance for the deployment of the web-app.

1. To initiate the instance launching, from Amazon EC2 console dashboard choose, launch instance.
2. Second step is to choose an AMI (Amazon Machine Image), this holds the information required for creating a new instance i.e. software requirement to act as a web server e.g. Windows, Linux, etc.
3. Next step includes choosing the correct instance type which includes some free as well as paid versions e.g. to remain under the free eligible tier using t2.micro or t3.micro is recommended.



**Figure 8: Steps to launch an instance in AWS**

1. After choosing the required tier, the instance must be configured where different parameters can be changed according to the requirement.
2. Storage option is provided next wherein as per the requirement it can be upgraded.
3. Also, tags can be added by providing keys and value combination.
4. Before reviewing the instance and launching it the configuration of security groups is done and a key pair is selected.

After creating and launching the EC2 Linux instance which is used in this project, it can be connected to the local computer using the SSH protocol. PuTTy gives this permission of connecting it to the local computer running windows. WinSCP is used for the file transfer to and from the machine’s server while PuTTY is used to interact with the server directly. Putty is just a command line interface to the local server while WinSCP acts as a file transfer application using Secure FTP.

# Conclusion

To conclude, a website has been deployed on AWS EC2 instance which predicts whether students will be placed in the campus placement based on their academic performance. To achieve this, a dataset containing various parameters was visualized, cleaned trained on different machine learning algorithm. Grid search cross-validation was used to find out which model gives the highest accuracy. The support vector classifier showed the highest accuracy of all the used algorithms and was used for training and testing the dataset. The dataset was divided into 80:20 ratio and the training accuracy achieved was 91.27% with the testing accuracy of 79.06%. Finally, the user interface was developed and the problem for this project and its related tasks were achieved.