BITS PILANI WILP

APPLIED MACHINE LEARNING

SSZG568

ASSIGNMENT-1

TITANIC SURVIVAL PREDICTION USING SKLEARN

NAME: PRAJWAL S TELKAR

BITS ID: 2023MT12205

MTech-Software Systems

April 7, 2024

**Introduction**

The Titanic dataset contains information about passengers aboard the Titanic, including whether they survived or not. In this report, we aim to use machine learning techniques to predict passenger survival based on the available data. Our goal is to develop a predictive model that accurately determines the likelihood of survival for each passenger. To achieve this, we will explore the dataset, preprocess the data, and select relevant features. We will then train and evaluate machine learning models, aiming to achieve a high level of accuracy while avoiding overfitting.

Through this analysis, we seek to gain insights into the factors that influenced survival on the Titanic and demonstrate the effectiveness of machine learning in predicting outcomes based on historical data.

**End To End ML Project Steps**

1. Look at the Big Picture:

* Define the objective of the project (e.g., predicting passenger survival on the Titanic).
* Determine how the machine learning model will be used in the broader context (e.g., decision support system for maritime safety).

1. Get the Data:

* Obtain the Titanic dataset, which contains information about passengers, including whether they survived or not.

1. Discover and Visualize the Data:

* Explore the dataset to understand its structure and contents.
* Use data visualization techniques to gain insights into the data (e.g., survival rates by gender or ticket class).

1. Prepare the Data for Machine Learning Algorithms:

* Clean the data by handling missing values and encoding categorical variables.
* Feature engineering: Create new features or transform existing ones to improve model performance.

1. Select a Model and Train It:

* Choose a machine learning model suitable for the task (e.g., Decision Tree Classifier).
* Split the data into training and testing sets.
* Train the model using the training data.

1. Fine-Tune Your Model:

* Use cross-validation to fine-tune hyperparameters and avoid overfitting.
* Evaluate the model on the test set to ensure generalization to new data.

1. Present Your Solution:

* Summarize the findings from the analysis.
* Present the model's performance metrics and any insights gained.

1. Launch, Monitor, and Maintain Your System:

* Deploy the model into a production environment if applicable.
* Monitor the model's performance over time and update it as needed to maintain its effectiveness.

**Get The Data**

**Step1: Load the dataset**

The first step is to load the dataset. We will be using the Titanic dataset, which contains information about passengers on the Titanic ship, including whether or not they survived.

A computer screen shot of a program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

**Data Visualization**

Data visualization helps us explore and communicate the patterns and trends in the dataset, which is essential for making informed decisions and building predictive models.

A screenshot of a computer

Description automatically generated

A graph of a distribution of passengers

Description automatically generatedA graph with numbers and lines

Description automatically generated

A graph showing a number of points

Description automatically generated with medium confidenceA graph of age distribution

Description automatically generated

A graph of a number of bars

Description automatically generated with medium confidenceA graph of a number of bars

Description automatically generated

A graph of a number of blue bars

Description automatically generatedA graph showing a distribution of sex

Description automatically generated

A bar graph with different colored squares

Description automatically generated

**Data Preparation**

In this step, we loaded the dataset and performed basic preprocessing steps to prepare the data for modeling. This included handling missing values and encoding categorical variables to convert them into a format suitable for machine learning algorithms.

**Step 1: Preprocess the dataset**

Before we can use the dataset for training our decision tree model, we need to preprocess it. First, we will handle missing values in the dataset. In this case, we will fill missing values in the ‘Age’ column with the median age, and missing values in the ‘Embarked’ column with the mode (most common value). Filling missing values with these central tendency measures helps to ensure that the datasets are complete and ready for further analysis and modeling.

A computer code with black text

Description automatically generated

**Step 2: Feature Selection/Engineering**

Feature engineering involves creating new features from existing ones to improve model performance. We didn't explicitly mention any new features, but this step could include creating interaction terms or transforming existing features to better represent the data.

Feature selection is the process of selecting the most relevant features for the model. In this case, we selected features such as 'Pclass' (passenger class), 'Sex', 'Age', 'SibSp' (number of siblings/spouses aboard), 'Parch' (number of parents/children aboard), 'Fare' (ticket fare), and 'Embarked' (port of embarkation).

A close up of words

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generated

**Split The Data into Training and Test Sets**

In the code, we import the train\_test\_split function from the sklearn.model\_selection module, which is used to split the dataset into training and testing sets. Here, we pass the X and y dataframes to the train\_test\_split function, which returns four dataframes: X\_train, X\_test, y\_train, and y\_test.

The test\_size parameter is set to 0.2, which means that 20% of the data is reserved for testing, and the remaining 80% is used for training. The random\_state parameter is set to 42 to ensure that the split is reproducible.

A screen shot of a computer

Description automatically generated

**Model Selection**

After we split the data into training and testing sets to train the models and evaluate their performance. The following models were trained on the training data and evaluated using cross-validation to estimate their accuracy.

1. Decision Tree Classifier
2. Random Forest

**1. Decision Tree Classification**

Decision Tree is a tree-like model where each internal node represents a feature or attribute, each branch represents a decision based on that feature, and each leaf node represents the outcome or target variable. Decision Tree Classifier is used to predict whether a passenger survived or not based on various features such as their age, gender, ticket class, fare, and embarkation port. The Decision Tree algorithm iteratively splits the dataset into subsets based on feature values to create a tree-like structure.

**Select the hyperparameters and tune the model**

Tuning the decision tree model is required to find the best hyperparameters that optimize the model's performance. The max\_depth parameter controls the maximum depth of the tree, which can help prevent overfitting. The min\_samples\_split parameter specifies the minimum number of samples required to split an internal node, while min\_samples\_leaf specifies the minimum number of samples required to be at a leaf node. The best hyper parameters were 'max\_depth': 3, 'max\_features': None, 'min\_samples\_leaf': 5, 'min\_samples\_split': 2

A screenshot of a computer program

Description automatically generated



**Model Evaluation and Cross Validation**

Cross-validation is a technique for evaluating machine learning models like the Decision Tree Classifier. It involves splitting the dataset into multiple subsets or folds, training the model on a subset, and testing it on a different subset. This process helps ensure that the model is robust and avoids overfitting. In our analysis, we conducted cross-validation with different fold numbers, including 5, 9, and 11. The best mean cross-validation accuracy was achieved with 11 folds, suggesting that this configuration provided the most reliable estimate of the model's performance. This outcome is likely due to the increased diversity in training and testing data combinations.

1. **First Pass:**

Set Cross Validation to 5

A screenshot of a computer

Description automatically generated

1. **Second Pass:**

Set Cross Validation to 9

A computer screen shot of a computer code

Description automatically generated

1. **Third Pass:**

Set Cross Validation to 11

A computer screen shot of a computer

Description automatically generated

**Visualize The Decision Tree Using Plot\_Tree**

A computer screen shot of a diagram

Description automatically generated

**2.Random Forest Classifier**

Random Forest Classifier is an ensemble learning method that constructs a multitude of decision trees during training and outputs the mode of the classes (classification) or the average prediction (regression) of the individual trees. It can handle a larger number of features and is less prone to overfitting compared to a single decision tree. The model can be trained using the same features as the Decision Tree Classifier and then evaluated based on accuracy and other relevant metrics.

**Select the hyperparameters and tune the model**

Similar to Decision Tree, we tune the hyper parameters and select the best one, max\_depth, n\_estimators, min\_samples\_split and min\_samples\_leaf hyper parameters are evaluated against the model and best hyper parameters are selected to tune the model. Here the best fit hyper parameters were ‘max\_depth': 10, ‘min\_samples\_leaf': 7, ‘min\_samples\_split': 2, ‘n\_estimators': 100

A screenshot of a computer code

Description automatically generated

**Model Evaluation and Cross Validation**

1. **First Pass:**

Set Cross Validation to 5

A white background with black text

Description automatically generated

1. **Second Pass:**

Set Cross Validation to 9

A screenshot of a computer

Description automatically generated

1. **Third Pass:**

Set Cross Validation to 11

A computer screen shot of a computer code

Description automatically generated

**Results**

**Comparing Decision Tree Classifier and Random Forest Classifier**

A close up of a computer screen

Description automatically generated

The Decision Tree Classifier achieved a mean cross-validation accuracy of 0.82042(~82.04%), while the Random Forest Classifier achieved a slightly higher mean accuracy of 0.82716(~82.71%). The difference in mean accuracy between the two models was 0.00673(~0.67%), indicating that the Random Forest model performed slightly better.

**Project Folder uploaded in Google Drive for reference**

* <https://drive.google.com/drive/folders/1V4SEO4llhNgAyHC5nc-ea6C_sdYMyd88?usp=drive_link>

**Predicted Values:**

1. **Decision Tree Model (CSV file)**

* [titanic\_predicted\_values\_dt\_classification.csv](https://drive.google.com/file/d/1yHjx5lk3P2WYWN5gUSWbo7GpMq2MyfKL/view?usp=drive_link)

1. **Random Classifier Model (CSV file)**

* [titanic\_predictions\_rf\_classification.csv](https://drive.google.com/file/d/1ZoModpRCcgzg3py-7XC_weF7ApzXvfw0/view?usp=drive_link)

**Code:**

* [2023mt12205\_ml\_assignment1.ipynb](https://drive.google.com/file/d/1RBOAcG356jBGvzRjby5mxNDLJBTKmSJE/view?usp=drive_link)

**Jupyter NoteBook PDF :**

* [2023mt12205-Assignment1-ML-JupyterCode.pdf](https://drive.google.com/file/d/19B_WlE1eOlCJts33zXmICzWGLlX2FPIq/view?usp=sharing)

**Source:**

* <https://www.kaggle.com/competitions/titanic/overview>
* <https://www.kaggle.com/competitions/titanic/data>

**References:**

* <https://www.kaggle.com/competitions/titanic/overview>
* <https://www.kaggle.com/competitions/titanic/data>
* [Implementing Decision Tree Algorithm for Classification with Titanic Dataset in Python | by Dr. Soumen Atta, Ph.D. | Medium](https://soumenatta.medium.com/implementing-decision-tree-algorithm-for-classification-with-titanic-dataset-in-python-dd508afccaa7)
* <https://www.geeksforgeeks.org/titanic-survival-prediction-using-tensorflow-in-python/>
* Aurelien Geron, “Hands-On Machine Learning with Scikit-Learn, Keras and Tensorflow”, O’Reilly, 2020

### [Supervised Machine Learning: Regression and Classification](https://www.coursera.org/learn/machine-learning?specialization=machine-learning-introduction) - Coursera