TECHNEX IIT (BHU) PROJECT REPORT ON DATA ANALYTICS AND MACHINE LEARNING

Submitted By Prajwal Satpute

Batch Code : OT20-DAT-024

<u>Submitted To</u>

Mr. Mayur Dev Sewak

General Manager, Operations

EISystems Services

Mr. Shailendra Singh

Trainer, ICT Domain,

Logicpro Inforsystems

EiSystems Services

ACKNOWLEDGMENT

It is great pleasure to present this report on “Data Analytics and Machine Learning” which I undertook as part of my B.E curriculum.

I am thankful to Technex and IIT (BHU) Varanasi for offering me such wonderful opportunity and I express my deep gratitude to Head of Technex for providing all possible help and assistance and encouragement.

It is pleasure that I find myself writing down these lines to express my sincere thanks to those who helped me for completing my project. I couldn’t find words to express my heartfelt gratitude for them.

First I would like to express my gratitude towards our trainer Mr. Shailendra Singh for placing complete faith and confidence in my ability to carry out this project and for providing me his time, inspiration, encouragement, and constant interest. I also thank him for his help in my every problem and providing good quality of course content.

The internship on “Data Analytics and Machine Learning” was very helpful to me giving the necessary background information and inspiration about the emerging field of Machine Learning and Data Analysis. Their contribution and technical support in preparing this report are greatly acknowledgeable.

Last but not least, I wish to thanks my parents for financing my studies for Technex as well as constantly encouraging me to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is grateful.

Index

❖ LIST OF FIGURES

SYSTEMS USED FOR PROJECT DEVELOPMENT

- Hardware Used :-
  - Intel(R) Core(TM) i3-7020U CPU @ 2.30 GHz 2.30 GHz
  - 8 GB RAM
  - NVIDIA GeForce MX110

- Software Used :-
  - Python Programming Language
  - Jupyter Notebook
  - Python Libraries – Numpy, Pandas, Scikit-Learn, Matplotlib, Seaborn

ABSTRACT

This report represents handwritten digit recognition on a very well-known datasets which are TITANIC(train) and IRIS using the classification models of supervised learning. The demand of text classification is growing significantly in web searching, data mining, web ranking, recommendation systems and so many other fields of information technology. This report illustrates the text classification process on TITANIC(train) dataset and classification of IRIS dataset using some standard supervised machine learning techniques. From the dataset, features have been extracted using various extracting methods. After this, to reduce the number of features of reducing the dimensions we have used SVM analysis for better performance for our proposed classifier. Then it has been trained by various classifiers. Then the accuracy score and confusion matrix of those classifiers has been shown in the code. Also some statistical analysis has been done for better understanding of the dataset. From those comparison, it has been shown that our proposed models has better accuracy and less error than other classifiers. The results are competitively compared to previous works and they provide a baseline for evaluation of future work. In this report, we investigate Decision Tree, Support-Vector Machines, and Logistic Regression classifiers implemented in Jupyter Notebook. The focus of our report is on comparing these classifiers by evaluating the classification accuracy and based on the size of training data sets. Accuracy is measured over correctly and incorrectly classified instances. Results obtained show that SVM and logistic outperforms with the high accuracy of more than 90% compared to other algorithms.

INTRODUCTION

Data is the new oil of the digital economy. We’re in a digital economy where data is more valuable than ever. It’s the key to the smooth functionality of everything from the government to local companies. Without it, progress would halt.

In the course we learned about a wide range skills spanning from gathering data, cleaning dataset to gather data which is of importance to our analysis, training our model to interpreting the data to gain valuable insights. The course is rightly named “Data Analytics and Machine Learning”.

Data Analytics is the science if analysing raw data in order to make conclusions about that information. It involved applying an algorithm or mechanical process to derive insights and for example, running through several datasets to look for meaningful correlation between each other. It is used in several industries to allow organizations and companies to make better decisions as well as verify and disprove existing theories and models.

Next, we learned about data mining. Data mining can be called as the subset of data analytics. Data mining is a systematic process of identifying and discovering hidden pattern and information in a dataset. It is known as data discovery in databases because it requires knowledge of machine learning, statistics and databases.

Now before we apply techniques of analysis and mining, we need to collect and manage data and the term that we used for this in our course is data warehousing, data warehousing is the process of collecting and managing data from varied sources. A data warehouse is typically used to connect and analyse business data from heterogeneous sources.

After collection and cleaning of data, the magical insights that we gain is with the help if machine learning. Machine learning is the study of computer algorithms that improve automatically through experience. The machine learning algorithms build a mathematical model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.

MOTIVATION

Hand written character recognition has been around since the 1980s. The task of handwritten digit recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognizing zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example - tax forms) and so on.

There are different challenges faced while attempting to solve this problem. The handwritten digits are not always of the same size, thickness, or orientation and position relative to the margins.

Our goal was to implement a pattern classification method to recognize the digits provided in TITANIC(train) dataset. The dataset used for our application is TITANIC(train) dataset composed of 891 training images. Each image is a 28 × 28 grauscale (0-255) labelled representation of an individual digit.

We have also used another dataset known as 'Iris' dataset which is a collection of petal length and width, sepal length and width of different species of iris flower. We took this dataset into consideration because we wanted to check how algorithms perform when they are given a smaller dataset.

METHODOLOGY

There are many different kinds of learning algorithms used in machine learning. Mainly there are two broad categories of learning algorithms – supervised and unsupervised learning algorithms. Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points.

In our project, we focused on three supervised learning algorithms which are Logistic Regression, Decision Tree and Support Vector Machine classifier. These algorithms are used for classification purposes. Let us look at them one by one.

Logistic Regression

The logistic regression algorithm is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. Logistic regression is a statistical model that, in its basic form, it uses a logistic function to model a binary dependent variable.. Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labelled “0” and “1”. The logistic function is a common S-shaped curve (sigmoid curve) with equation-

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

where x_0= the *x* value of the sigmoid’s midpoint,
L= the curve’s maximum value
k= the logistic growth or steepness of curve.

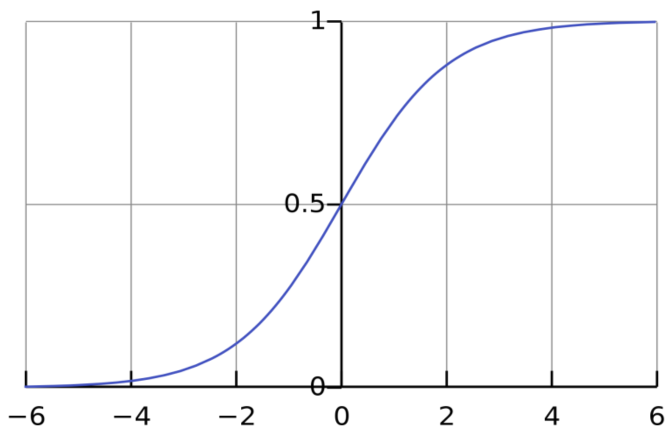


Fig.1 - Standard Logistic Sigmoid Function, i.e., L=1, k=1, x0 = 0

Logistic Regression measures the realtionship between catgorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative distribution function of logistic disrtibution. Although from name we may infer that logistic regression is a regression algorithm like linear regression but it is a classication algorithm.

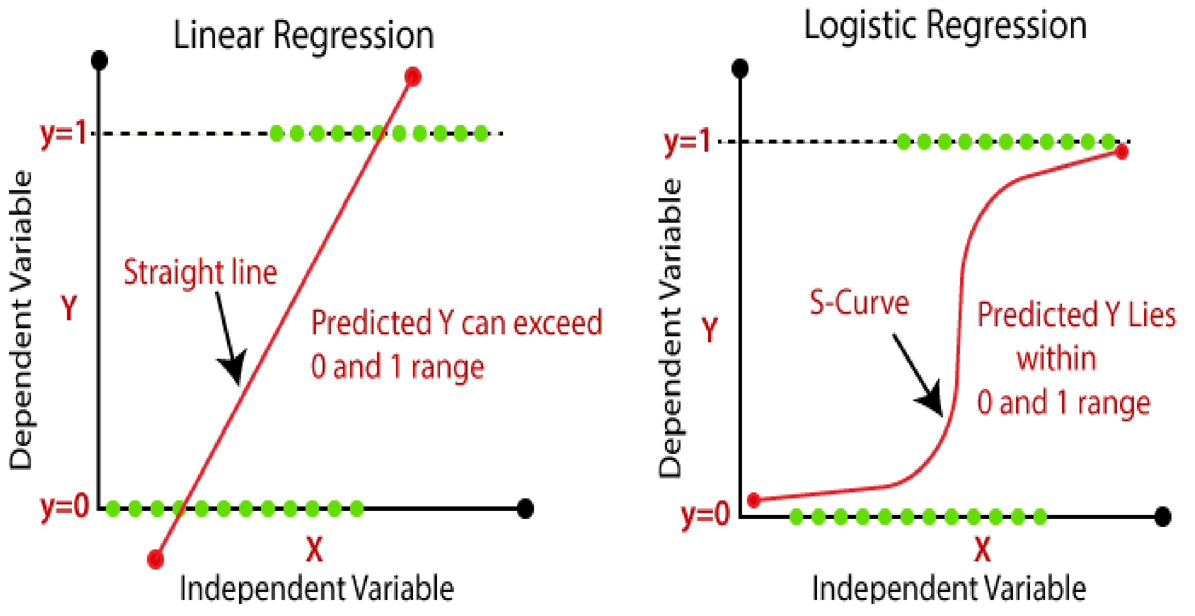


Fig.2 - Linear regression vs Logistic regression

There are many extensions of logistic regression algorithm. Multinomial logistic regression handles the case of multi-way categorical dependent variable.

Decision Tree

In machine learning, a decision tree is used (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves).

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

Below are some of the assumptions we make while using Decision tree:

- In the beginning, the whole training set is considered as the root.
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are distributed recursively on the basis of attribute values.
- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

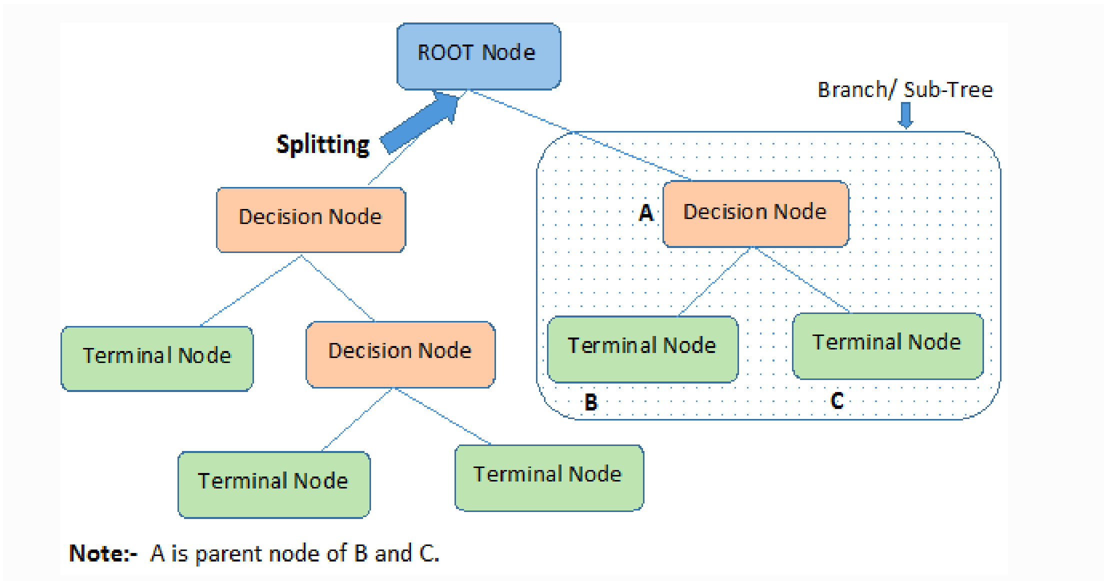


Fig.3 - Splitting of root node into decision node resulting in the formation of decision tree

Support Vector Machine

Support-Vector Machines or SVMs are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. If we are given some data points each belong to one of two classes, and the goal is to decide which class a 'new' data point will be in then we may use SVM. SVM views a data point as a p-dimensional vector (a list of p numbers) and we want to know whether we can separate such points with a (p-1)-dimensional hyperplane. This is called linear classifier.

Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. We choose that hyperplane which represents the largest separation between two classes. This means that the distance from the hyperplane to the nearest data point on each side is maximized. This type of hyperplane is called as maximum-margin hyperplane.

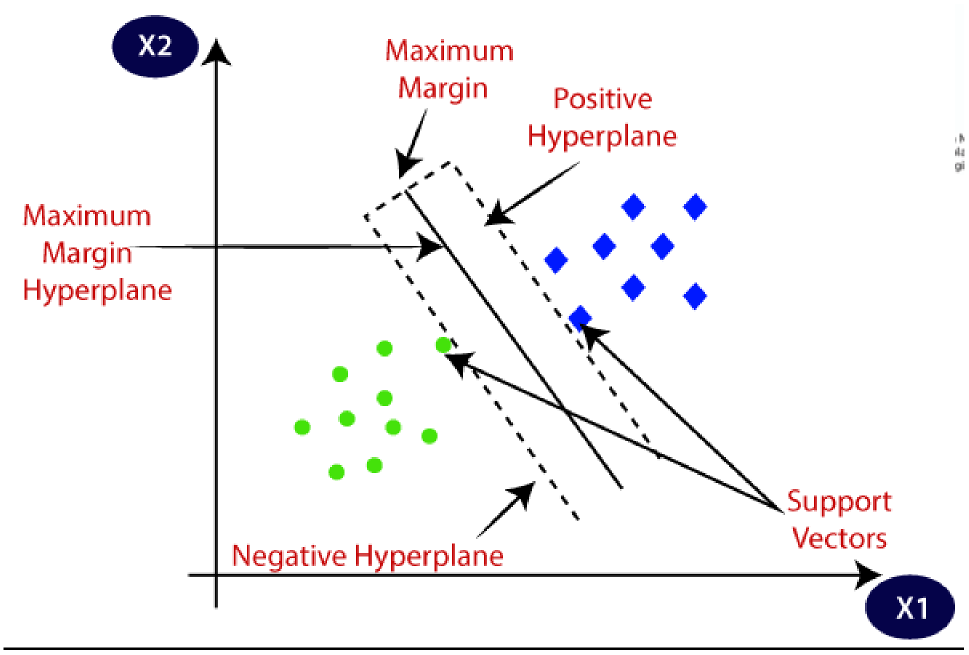


Fig.4 - SVM showing classification of target as 'Yes' or 'No'

SVM algorithms use a set of mathematical functions known as kernel. The function of a kernel is to take data as input and transform it into the required form. The kernel functions return the inner product between two points in a suitable feature space. A feature is an individual measurable property or characteristic of a phenomenon being observed and these features are represented as feature vectors. The vector space associated with these vectors is often called the feature space. Kernel functions reduce computational cost in very high-dimensional spaces.

INFORMATION ABOUT DATASET

TITANIC(train) Dataset

The principal source for data about Titanic passengers is the Encyclopedia Titanica. The datasets used here were begun by a variety of researchers. One of the original sources is Eaton & Haas (1994) Titanic: Triumph and Tragedy, Patrick Stephens Ltd, which includes a passenger list created by many researchers and edited by Michael A. Findlay. The dataset contains 12 columns. They are PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked. Dataset contains 891 entries.

IRIS Dataset

The Iris dataset was used in R.A. Fisher's classic 1936 paper, The Use of Multiple Measurements in Taxonomic Problems, and can also be found on the UCI Machine Learning Repository. It includes three iris species with 50 samples each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly separable from each other. There are 6 columns in this dataset. They are Id, sepal length in cm, sepal width in cm, petal length in cm, petal width in cm, and species. This dataset is sometimes called as the “Hello, World!” dataset of data analytics.

MACHINE LEARNING CODE

Logistic Regression Code

IMPORT LIBRARIES FOR DATA FIXING & EDA

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

GET DATA

In [2]:
```
train_df = pd.read_csv('train.csv')
```

In [4]:
```
# Let's check head,info & describe for train as we'll be fixing data & doing EDA for train_df first

train_df.head()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

EXPLORATORY DATA ANALYSIS (EDA) FOR train DATA

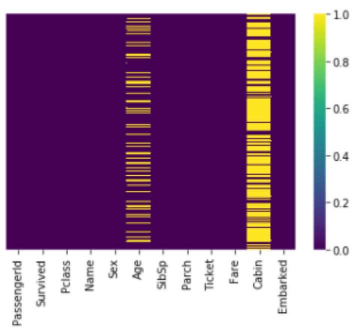
FOR Missing VALUES

In [45]:
```
train_df.isnull().sum()
```

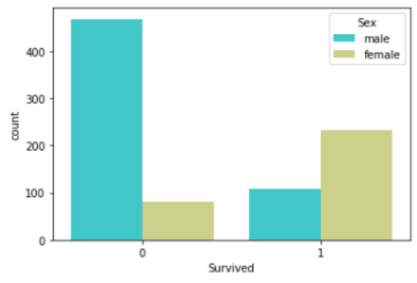
Out[45]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [46]:
```
sns.heatmap(train_df.isnull(),cmap='viridis',yticklabels=False)
```

Out[46]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x16565b4b9c8>
```

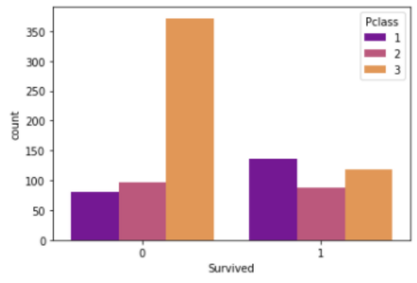


```
In [47]: sns.countplot(x='Survived',data=train_df,hue='Sex',palette='rainbow')
Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x1656532b088>
```

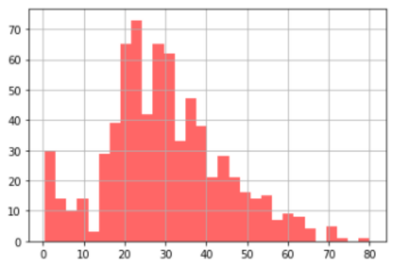


```
In [48]: train_df[train_df['Survived']==1]['Survived'].sum()
Out[48]: 342
```

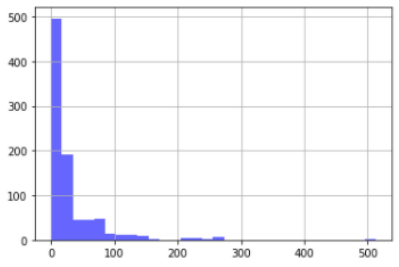
```
In [49]: sns.countplot(x='Survived',data=train_df,hue='Pclass',palette='plasma')
Out[49]: <matplotlib.axes._subplots.AxesSubplot at 0x16565c64248>
```

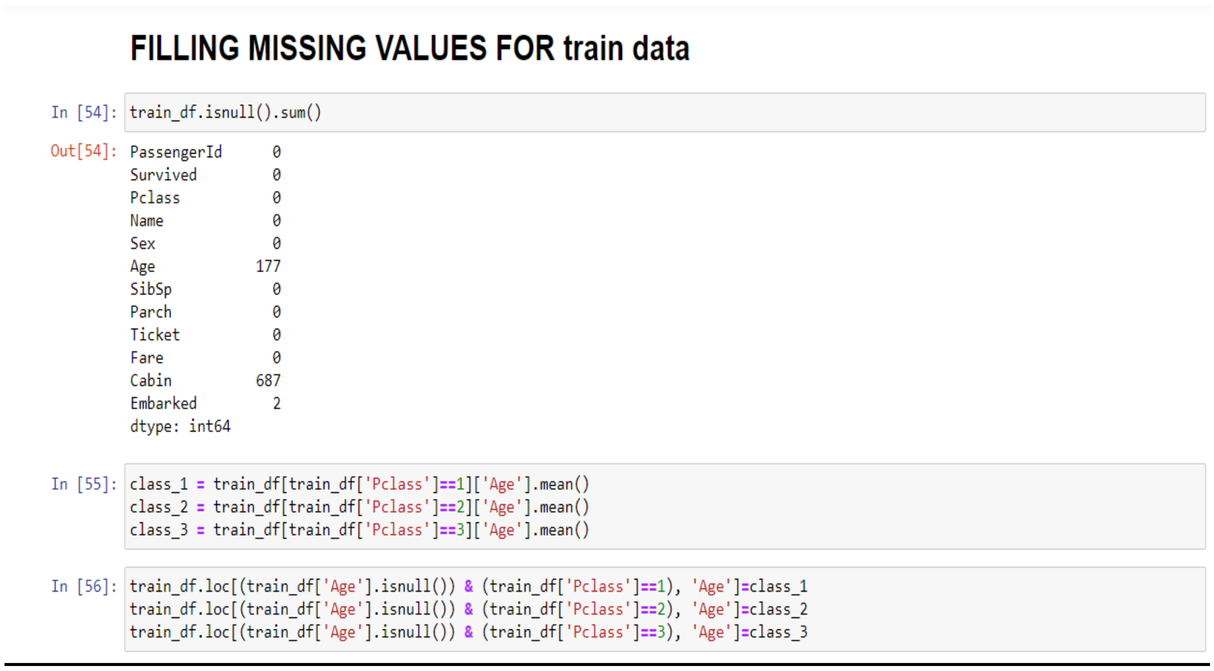


```
In [50]: train_df['Age'].hist(bins=30,color='red',alpha=0.6)
Out[50]: <matplotlib.axes._subplots.AxesSubplot at 0x16565cdd848>
```



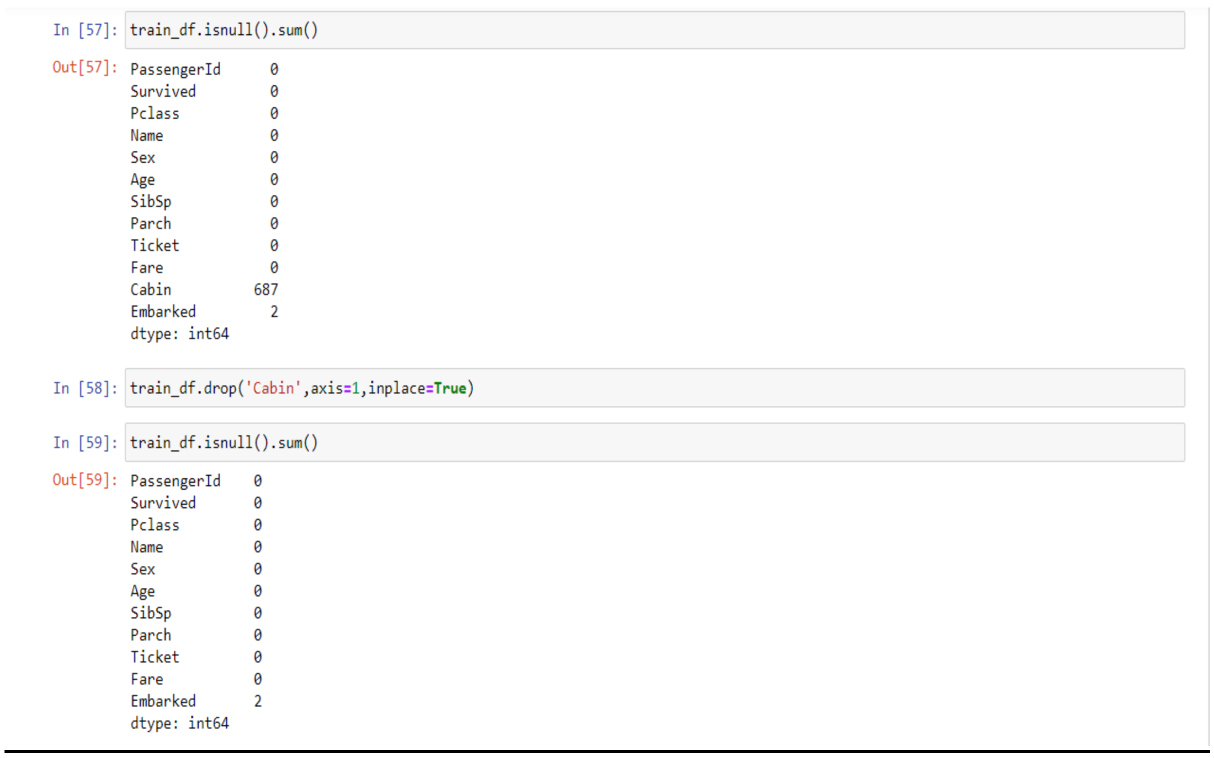
```
In [51]: train_df['Fare'].hist(bins=30,color='blue',alpha=0.6)
Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x16565db6a48>
```



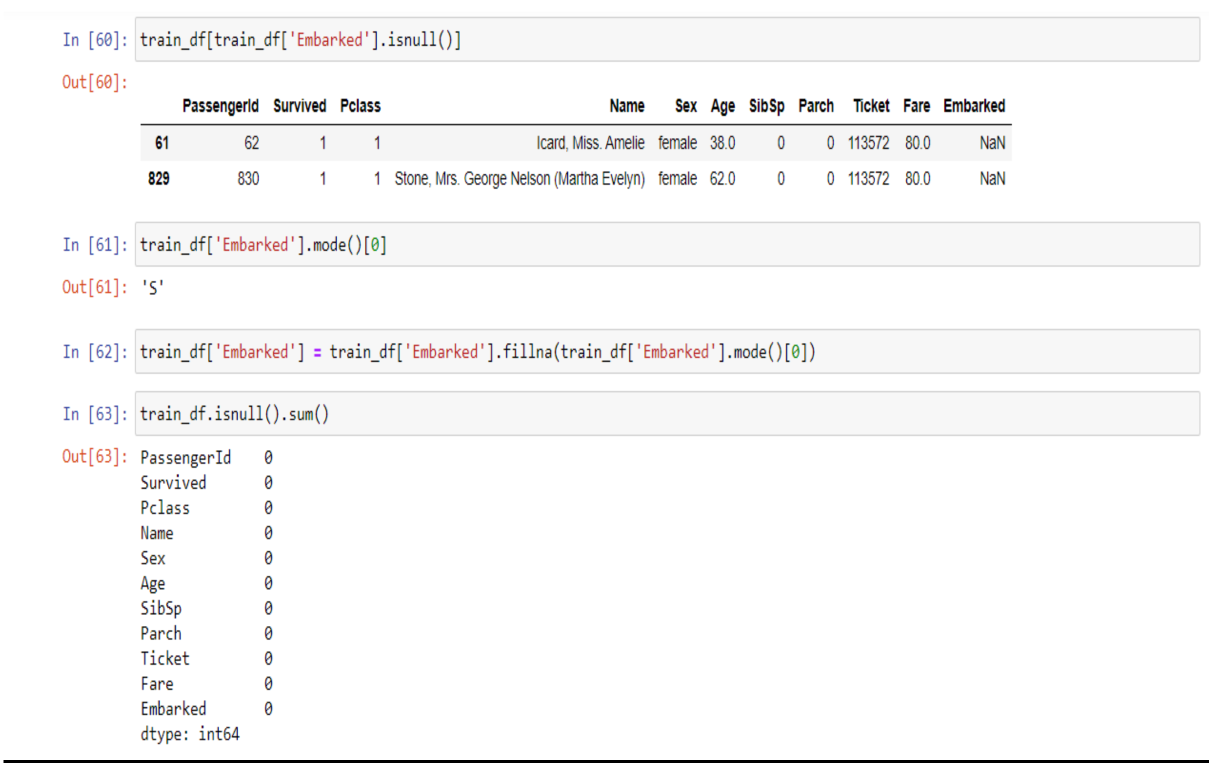


FILLING MISSING VALUES FOR train data

```
In [54]: train_df.isnull().sum()

Out[54]: PassengerId      0
         Survived         0
         Pclass           0
         Name             0
         Sex              0
         Age            177
         SibSp            0
         Parch            0
         Ticket           0
         Fare             0
         Cabin          687
         Embarked         2
         dtype: int64

In [55]: class_1 = train_df[train_df['Pclass']==1]['Age'].mean()
         class_2 = train_df[train_df['Pclass']==2]['Age'].mean()
         class_3 = train_df[train_df['Pclass']==3]['Age'].mean()

In [56]: train_df.loc[(train_df['Age'].isnull()) & (train_df['Pclass']==1), 'Age']=class_1
         train_df.loc[(train_df['Age'].isnull()) & (train_df['Pclass']==2), 'Age']=class_2
         train_df.loc[(train_df['Age'].isnull()) & (train_df['Pclass']==3), 'Age']=class_3
```



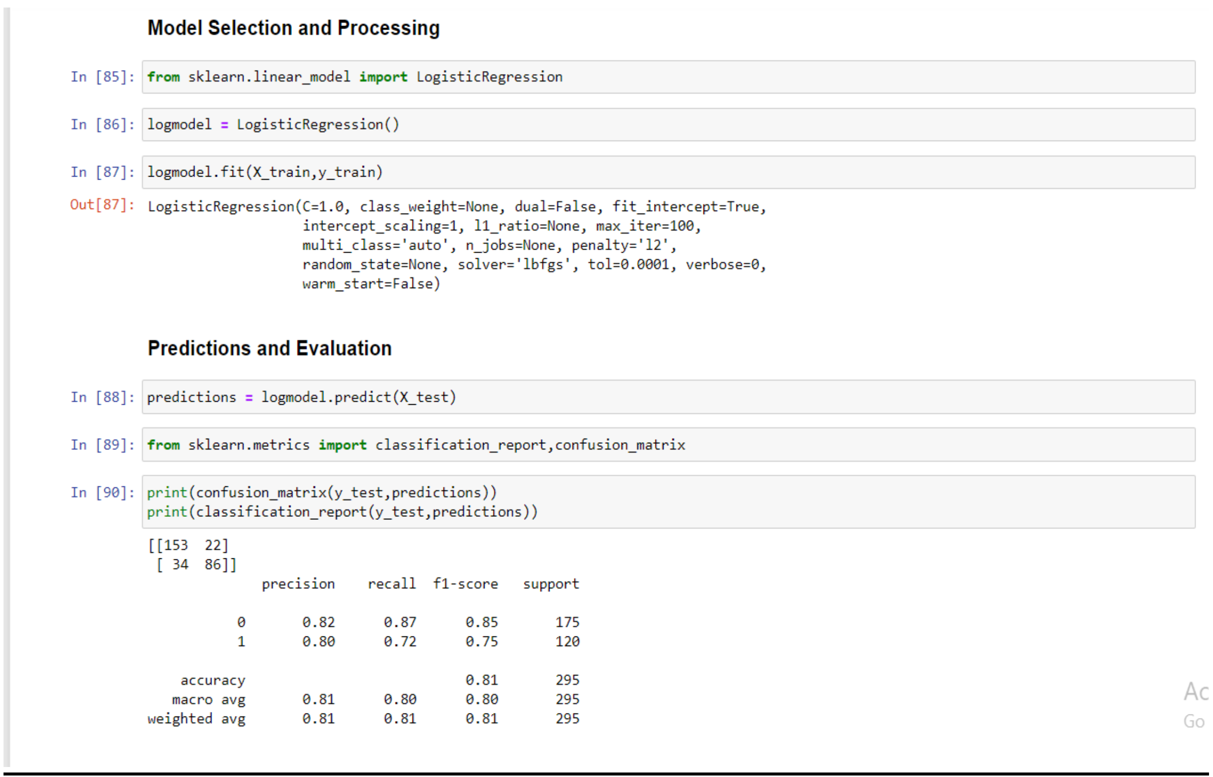
```
In [57]: train_df.isnull().sum()

Out[57]: PassengerId      0
         Survived         0
         Pclass           0
         Name             0
         Sex              0
         Age              0
         SibSp            0
         Parch            0
         Ticket           0
         Fare             0
         Cabin          687
         Embarked         2
         dtype: int64

In [58]: train_df.drop('Cabin',axis=1,inplace=True)

In [59]: train_df.isnull().sum()

Out[59]: PassengerId    0
         Survived       0
         Pclass         0
         Name           0
         Sex            0
         Age            0
         SibSp          0
         Parch          0
         Ticket         0
         Fare           0
         Embarked       2
         dtype: int64
```



```
In [60]: train_df[train_df['Embarked'].isnull()]
Out[60]:
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
61	62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	NaN
829	830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	NaN

```
In [61]: train_df['Embarked'].mode()[0]

Out[61]: 'S'

In [62]: train_df['Embarked'] = train_df['Embarked'].fillna(train_df['Embarked'].mode()[0])

In [63]: train_df.isnull().sum()

Out[63]: PassengerId    0
         Survived       0
         Pclass         0
         Name           0
         Sex            0
         Age            0
         SibSp          0
         Parch          0
         Ticket         0
         Fare           0
         Embarked       0
         dtype: int64
```



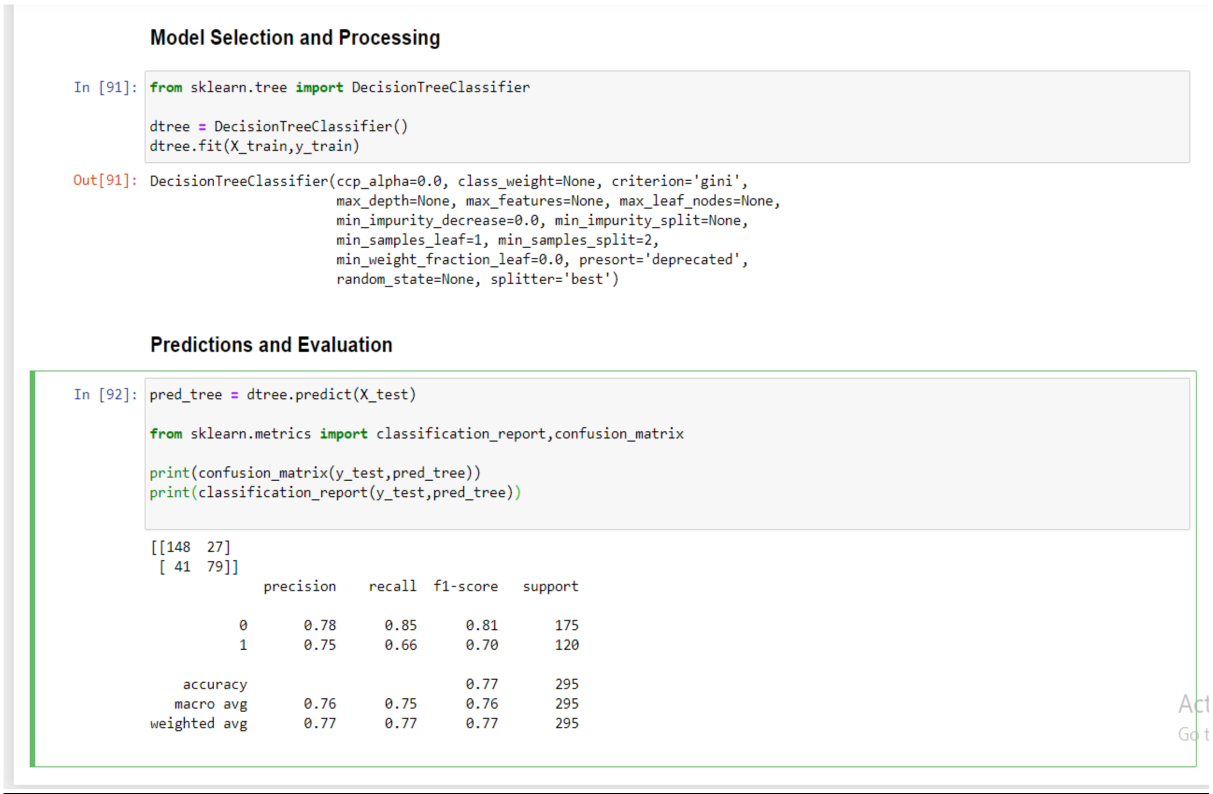
Model Selection and Processing

```
In [85]: from sklearn.linear_model import LogisticRegression

In [86]: logmodel = LogisticRegression()

In [87]: logmodel.fit(X_train,y_train)

Out[87]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                            intercept_scaling=1, l1_ratio=None, max_iter=100,
                            multi_class='auto', n_jobs=None, penalty='l2',
                            random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                            warm_start=False)
```

Predictions and Evaluation

```
In [88]: predictions = logmodel.predict(X_test)

In [89]: from sklearn.metrics import classification_report,confusion_matrix

In [90]: print(confusion_matrix(y_test,predictions))
         print(classification_report(y_test,predictions))

         [[153  22]
          [ 34  86]]
                       precision    recall  f1-score   support

                    0       0.82      0.87      0.85       175
                    1       0.80      0.72      0.75       120

             accuracy                           0.81       295
            macro avg       0.81      0.80      0.80       295
         weighted avg       0.81      0.81      0.81       295
```

Decision Tree



Model Selection and Processing

```
In [91]: from sklearn.tree import DecisionTreeClassifier

         dtree = DecisionTreeClassifier()
         dtree.fit(X_train,y_train)

Out[91]: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                                max_depth=None, max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort='deprecated',
                                random_state=None, splitter='best')
```

Predictions and Evaluation

```
In [92]: pred_tree = dtree.predict(X_test)

         from sklearn.metrics import classification_report,confusion_matrix

         print(confusion_matrix(y_test,pred_tree))
         print(classification_report(y_test,pred_tree))

[[148  27]
 [ 41  79]]
              precision    recall  f1-score   support

           0       0.78      0.85      0.81       175
           1       0.75      0.66      0.70       120

    accuracy                           0.77       295
   macro avg       0.76      0.75      0.76       295
weighted avg       0.77      0.77      0.77       295
```

Support Vector Machine

```
In [1]: import seaborn as sns
```

```
In [2]: iris = sns.load_dataset('iris')
        iris
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

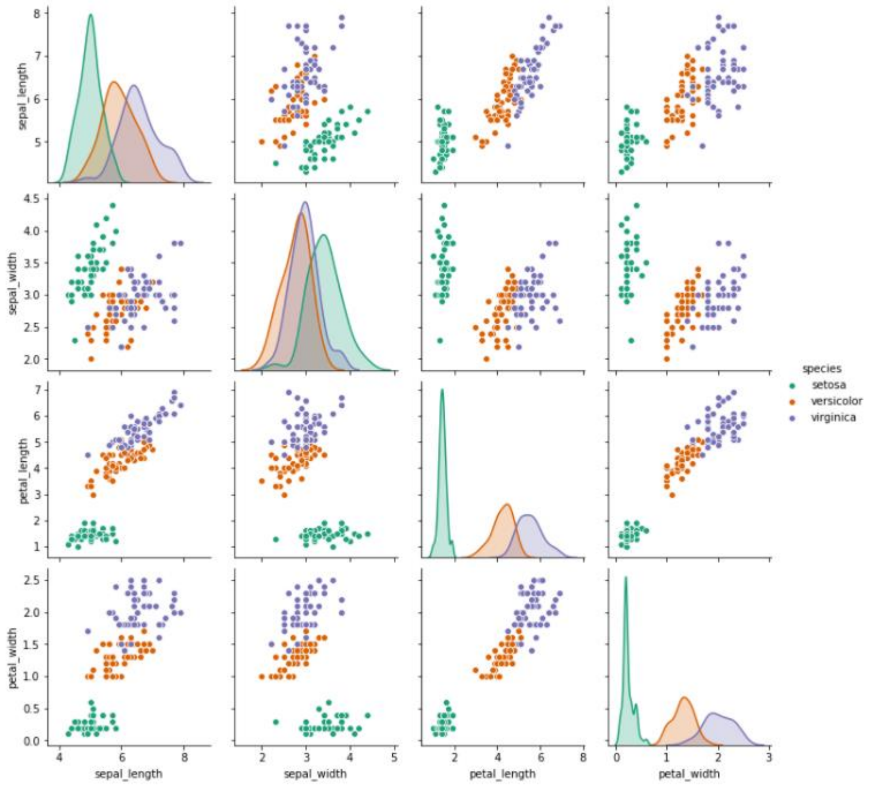
Exploratory Data Analysis

Time to put your data viz skills to the test! Try to recreate the following plots, make sure to import the libraries you'll need!

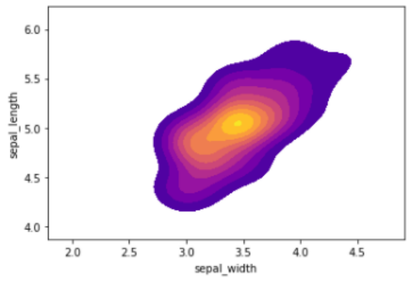
Import some libraries you think you'll need.

```
In [3]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
```

```
In [4]: sns.pairplot(iris,hue='species',palette='Dark2')
Out[4]: <seaborn.axisgrid.PairGrid at 0x289848e9e88>
```



```
In [5]: setosa = iris[iris['species']=='setosa']
        sns.kdeplot(setosa['sepal_width'],setosa['sepal_length'],cmap='plasma',shade=True,shade_lowest=False)

Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x28985905ac8>
```



Spliting Data for Model

```
In [6]: from sklearn.model_selection import train_test_split

In [7]: X = iris.drop('species',axis=1)
        y=iris['species']
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
```

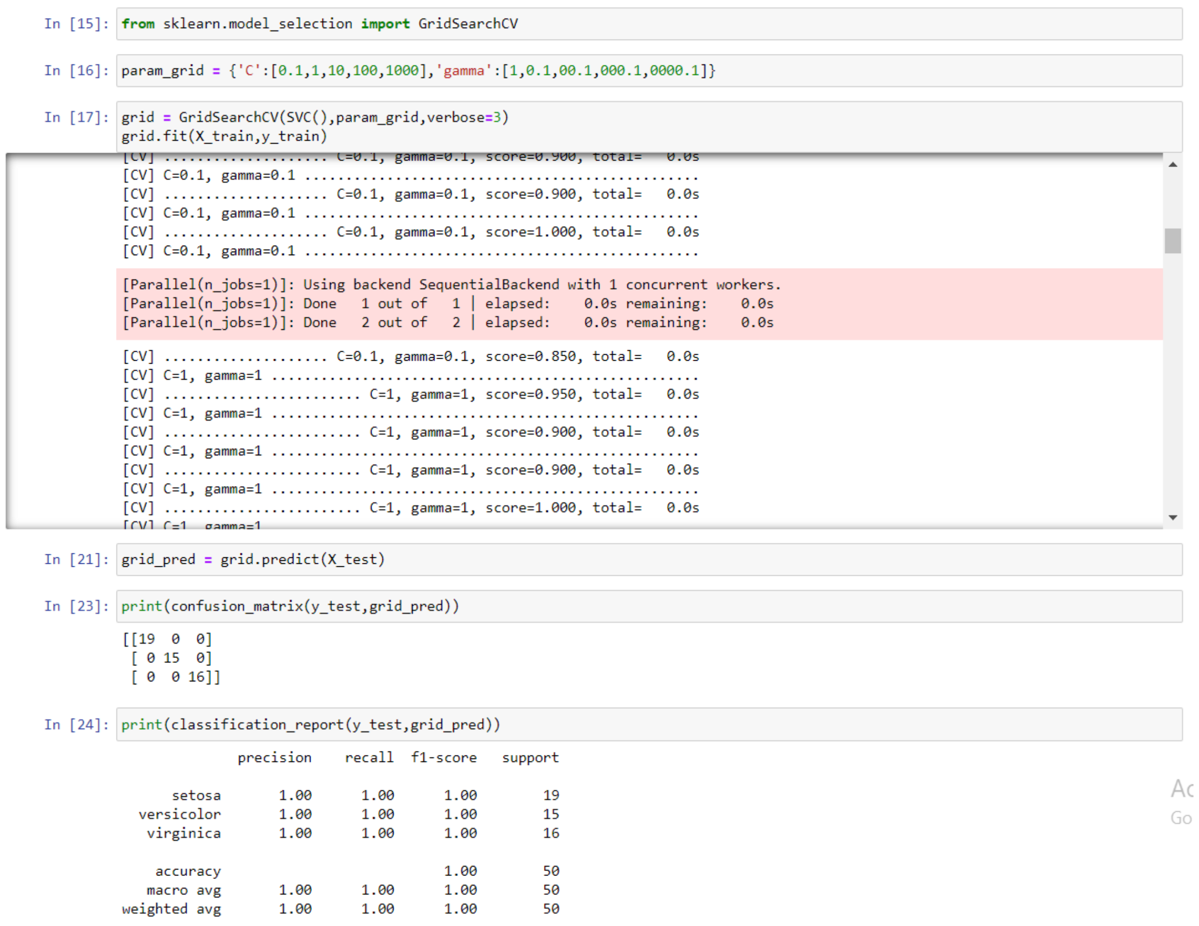
Model Selection and Processing

```
In [23]: from sklearn.svm import SVC

In [24]: model = SVC()

In [25]: model.fit(X_train,y_train)

Out[25]: SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
             decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
             max_iter=-1, probability=False, random_state=None, shrinking=True,
             tol=0.001, verbose=False)
```

Model Predictions and Evaluation

```
In [26]: pred = model.predict(X_test)

In [27]: from sklearn.metrics import classification_report,confusion_matrix

In [28]: print(confusion_matrix(y_test,pred))

         [[19  0  0]
          [ 0 15  0]
          [ 0  0 16]]

In [29]: print(classification_report(y_test,pred))

                       precision    recall  f1-score   support

               setosa       1.00      1.00      1.00        19
           versicolor       1.00      1.00      1.00        15
            virginica       1.00      1.00      1.00        16

             accuracy                           1.00        50
            macro avg       1.00      1.00      1.00        50
         weighted avg       1.00      1.00      1.00        50
```

SVM using Gridsearch



```
In [15]: from sklearn.model_selection import GridSearchCV

In [16]: param_grid = {'C':[0.1,1,10,100,1000],'gamma':[1,0.1,00.1,000.1,0000.1]}

In [17]: grid = GridSearchCV(SVC(),param_grid,verbose=3)
         grid.fit(X_train,y_train)
```

```
[CV] .................... C=0.1, gamma=0.1, score=0.900, total=   0.0s
[CV] C=0.1, gamma=0.1 ................................................
[CV] .................... C=0.1, gamma=0.1, score=0.900, total=   0.0s
[CV] C=0.1, gamma=0.1 ................................................
[CV] .................... C=0.1, gamma=0.1, score=1.000, total=   0.0s
[CV] C=0.1, gamma=0.1 ................................................

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s remaining:    0.0s

[CV] .................... C=0.1, gamma=0.1, score=0.850, total=   0.0s
[CV] C=1, gamma=1 ....................................................
[CV] ........................ C=1, gamma=1, score=0.950, total=   0.0s
[CV] C=1, gamma=1 ....................................................
[CV] ........................ C=1, gamma=1, score=0.900, total=   0.0s
[CV] C=1, gamma=1 ....................................................
[CV] ........................ C=1, gamma=1, score=0.900, total=   0.0s
[CV] C=1, gamma=1 ....................................................
[CV] ........................ C=1, gamma=1, score=1.000, total=   0.0s
[CV] C=1, gamma=1
```

```
In [21]: grid_pred = grid.predict(X_test)

In [23]: print(confusion_matrix(y_test,grid_pred))

         [[19  0  0]
          [ 0 15  0]
          [ 0  0 16]]

In [24]: print(classification_report(y_test,grid_pred))

                       precision    recall  f1-score   support

              setosa       1.00      1.00      1.00        19
          versicolor       1.00      1.00      1.00        15
           virginica       1.00      1.00      1.00        16

            accuracy                           1.00        50
           macro avg       1.00      1.00      1.00        50
        weighted avg       1.00      1.00      1.00        50
```

PROJECT SUMMARY

In this machine learning project, we took open source datasets that are publicly available and discussed various classification algorithms of supervised learning. With the use of different classification algorithms and after comparing their accuracy, a program can be built to recognize the handwritten digits with 81% accuracy. The accuracy rate can be higher based on the chosen machine learning algorithm.

I used Scikit-Learn implementation of Logistic Regression and Decision Tree algorithm for TITANIC(train) dataset and SVM algorithm for IRIS dataset. I trained my model by giving it 77% of data in dataset after cleaning it using Numpy and Pandas. I found that classification algorithms used in this project have useful applications. By doing this project, I have practiced what I have learned in the “Data Analytics and Machine Learning” course. My program probably does not beat the state of art in digit recognition and classification. However, I have for the first time observed the hands–on application of using the classification algorithms of supervised learning. This experience will definitely be helpful for my future study and research.

FUTURE SCOPE

Machine Learning is the latest technology and widely accepted nowadays and research work is still done in this. There are far reaching implications of the current research work. Projects like this kindle the curiosity of individuals to do research work.

Future scope of machine learning is quite bright. This project has its fair share of future possibilities such as:-

- Decision Tree is used in classifying the data this project. This can be extended to Random Forest Classifier algorithm as it uses multitudes of decision tree to arrive at exact decision which increases the recognition capability of our model manifolds.
- In the area of supervised learning, there are many algorithms which we haven't covered like Naïve Bayes algorithm. Bayes classifier are also known for good accuracy because they work on probability of the occurrence of the event.
- Machine Learning is the subset of Artificial Intelligence but machine learning also has a subset which is called as deep learning. Deep learning introduces the fascinating concept of neural network. Using neural networks, about 99% accuracy has been achieved for many datasets. This project work can be extended to include deep learning which will definitely increase accuracy of our models.

REFERENCES

- www.google.com
- www.wikipedia.org
- www.javatpoint.com
- www.wikimedia.org
- www.kaggle.com