IPL Winner Prediction Using Classification Algorithm

Importing the numpy, matplotlib, pandas and seaborn packages

In [55]:

```
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
import seaborn as sn
```

Loading the dataset

In [56]:

```
ipl_matches = pd.read_csv("C:/MBA/Business Analytics/DataSet/IPL Matches 2008-2020.csv")
```

Length of dataset

In [57]:

```
len(ipl_matches)
```

Out[57]:

```
816
```

In [63]:

```
ipl_matches.types
```

Out[63]:

```
id                   int64
city                object
date                object
player_of_match     object
venue               object
neutral_venue        int64
team1               object
team2               object
toss_winner         object
toss_decision       object
winner              object
result              object
result_margin      float64
eliminator          object
method              object
umpire1             object
umpire2             object
dtype: object
```

Collumns in the entire IPL dataset

In [61]:

```
print(ipl_matches.columns.values.tolist())
```

```
['id', 'city', 'date', 'player_of_match', 'venue', 'neutral_venue', 'team1', 'team2', 'toss_winner', 'toss_decision',
'winner', 'result', 'result_margin', 'eliminator', 'method', 'umpire1', 'umpire2']
```

Converting the date to int format

In [33]:

```
ipl_matches['season'] = ipl_matches['date'].str[:4].astype(int)
```

Replacing the old team and venues names with their current names

In [34]:

```
ipl_matches.replace(to_replace ="Deccan Chargers", value ="Sunrisers Hyderabad",inplace=True)
ipl_matches.replace(to_replace ="Bangalore", value ="Bengaluru",inplace=True)
ipl_matches.replace(to_replace ="Rising Pune Supergiant", value ="Rising Pune Supergiants",inplace=True)
ipl_matches.replace(to_replace ="Pune Warriors", value ="Rising Pune Supergiants",inplace=True)
ipl_matches.replace(to_replace ="M Chinnaswamy Stadium", value ="M.Chinnaswamy Stadium",inplace=True)
ipl_matches.replace(to_replace ="Subrata Roy Sahara Stadium", value ="Maharashtra Cricket Association Stadium",inplace=T
ipl_matches.replace(to_replace ="Delhi Daredevils", value ="Delhi Capitals",inplace=True)
ipl_matches.replace(to_replace ="Punjab Cricket Association IS Bindra Stadium, Mohali", value ="Punjab Cricket Associati
```

Encoding the team names, citiy names and toss decesion to numeric value

In [35]:

```
team_name = {'Chennai Super Kings' : 1, 'Rajasthan Royals' : 2, 'Kolkata Knight Riders' : 3, 'Kings XI Punjab' : 4,
             'Delhi Capitals' : 5, 'Royal Challengers Bangalore' : 6, 'Mumbai Indians' : 7, 'Sunrisers Hyderabad' : 8,
             'Kochi Tuskers Kerala' : 9, 'Rising Pune Supergiants' : 10, 'Gujarat Lions' : 11 }
decesion = {'field': 0, 'bat': 1}
city = {'Bengaluru':1, 'Chandigarh':2, 'Delhi':3, 'Mumbai':4, 'Kolkata':5, 'Jaipur':6,
       'Hyderabad':7, 'Chennai':8, 'Cape Town':9, 'Port Elizabeth':10, 'Durban':11,
       'Centurion':12, 'East London':13, 'Johannesburg':14, 'Kimberley':15,
       'Bloemfontein':16, 'Ahmedabad':17, 'Cuttack':18, 'Nagpur':19, 'Dharamsala':20,
       'Kochi':21, 'Indore':22, 'Visakhapatnam':23, 'Pune':24, 'Raipur':25, 'Ranchi':26,
       'Abu Dhabi':27, 'Rajkot':28, 'Kanpur':29, 'Dubai':30, 'Sharjah':31}
```

Replacing the ecoded values and removing the nan (not a number) values

In [36]:

```
ipl_matches.team1 = [team_name[item] for item in ipl_matches.team1]
ipl_matches.team2 = [team_name[item] for item in ipl_matches.team2]
ipl_matches.toss_winner = [team_name[item] for item in ipl_matches.toss_winner]
ipl_matches.toss_decision = [decesion[item] for item in ipl_matches.toss_decision]

ipl_matches = ipl_matches.replace('',nm.nan)
ipl_matches = ipl_matches[ipl_matches['winner'].notna()]
ipl_matches = ipl_matches[ipl_matches['city'].notna()]
ipl_matches.winner = [team_name[item] for item in ipl_matches.winner]
```

Length of rows after pre processing of data

In [37]:

```
len(ipl_matches)
```

Out[37]:

```
799
```

Structure of Data Frame

In [38]:

```
ipl_matches.describe()
```

Out[38]:

	id	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result_margin	season
count	7.990000e+02	799.000000	799.000000	799.000000	799.000000	799.000000	799.000000	786.000000	799.000000
mean	7.563968e+05	0.080100	5.047559	5.007509	4.937422	0.392991	4.886108	17.418575	2013.919900
std	3.086008e+05	0.271618	2.632026	2.636726	2.657771	0.488721	2.622592	22.120149	3.697671
min	3.359820e+05	0.000000	1.000000	1.000000	1.000000	0.000000	1.000000	1.000000	2008.000000
25%	5.012235e+05	0.000000	3.000000	3.000000	3.000000	0.000000	3.000000	6.000000	2011.000000
50%	7.292850e+05	0.000000	5.000000	5.000000	5.000000	0.000000	5.000000	8.000000	2014.000000
75%	1.082630e+06	0.000000	7.000000	7.000000	7.000000	1.000000	7.000000	20.000000	2017.000000
max	1.237181e+06	1.000000	11.000000	11.000000	11.000000	1.000000	11.000000	146.000000	2020.000000

Attribute selection

In [39]:
```
ipl_matches = ipl_matches[['team1','team2','city','toss_decision','toss_winner','venue','winner']]
X = ipl_matches.iloc[:, [0,1,3,4]].values
ipl_matches.head(2)
```

Out[39]:

	team1	team2	city	toss_decision	toss_winner	venue	winner
0	6	3	Bengaluru	0	6	M.Chinnaswamy Stadium	3
1	4	1	Chandigarh	1	1	Punjab Cricket Association Stadium, Mohali	1

Structure after attribute selection

In [40]:
```
df = pd.DataFrame(ipl_matches)
df.describe()
```

Out[40]:

	team1	team2	toss_decision	toss_winner	winner
count	799.000000	799.000000	799.000000	799.000000	799.000000
mean	5.047559	5.007509	0.392991	4.937422	4.886108
std	2.632026	2.636726	0.488721	2.657771	2.622592
min	1.000000	1.000000	0.000000	1.000000	1.000000
25%	3.000000	3.000000	0.000000	3.000000	3.000000
50%	5.000000	5.000000	0.000000	5.000000	5.000000
75%	7.000000	7.000000	1.000000	7.000000	7.000000
max	11.000000	11.000000	1.000000	11.000000	11.000000

Checking if there is any null value in any collumn

In [41]:

```
df[pd.isnull(df['city'])]
```

Out[41]:

team1	team2	city	toss_decision	toss_winner	venue	winner

Making a predictive model

In [42]:

```
from sklearn.preprocessing import LabelEncoder
var_mod = ['city','toss_decision','venue']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i])
df.dtypes
```

Out[42]:

```
team1            int64
team2            int64
city             int32
toss_decision    int32
toss_winner      int64
venue            int32
winner           int64
dtype: object
```

Impporting modules from sklearn

In [43]:

```
from sklearn.model_selection import KFold  #For K-fold cross validation
from sklearn.model_selection import StratifiedKFold
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn import metrics
```

Generic function for calculating classification model accuracy and cross validation percentage

In [44]:

```
def classification_method(model_type, data, predictors, outcome):
  model_type.fit(data[predictors],data[outcome])
  predictions = model_type.predict(data[predictors])
  accuracy = metrics.accuracy_score(predictions,data[outcome])
  print('Accuracy : %s' % '{0:.3%}'.format(accuracy))
  kf = KFold(n_splits=2, shuffle=False)
 # kf = KFold(data.shape[0], n_folds=7)
  error = []
  for train, test in kf.split(X):
    train_predictors = (data[predictors].iloc[train,:])
    train_target = data[outcome].iloc[train]
    model_type.fit(train_predictors, train_target)
    error.append(model_type.score(data[predictors].iloc[test,:], data[outcome].iloc[test]))

  print('Cross-Validation Score : %s' % '{0:.3%}'.format(nm.mean(error)))

  model_type.fit(data[predictors],data[outcome])
```

Calculating results using Logistic Classification Algorithm

In [45]:

```
from sklearn.linear_model import LogisticRegression
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
model_type = LogisticRegression()
classification_method(model_type,df,predictor_var,outcome_var)
```

```
c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConv
ersionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for exampl
e using ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\linear_model\_logistic.py:763: C
onvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessin
g.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/m
odules/linear_model.html#logistic-regression)

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConv
ersionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for exampl
e using ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\linear_model\_logistic.py:763: C
onvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessin
g.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/m
odules/linear_model.html#logistic-regression)

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConv
ersionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for exampl
e using ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\linear_model\_logistic.py:763: C
onvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessin
g.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/m
odules/linear_model.html#logistic-regression)
```

```
Accuracy : 35.544%
Cross-Validation Score : 29.664%
```

```
c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConv
ersionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for exampl
e using ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\linear_model\_logistic.py:763: C
onvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessin
g.html)
Please also refer to the documentation for alternative solver options:
```

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)

Calculating results using Random Forest Classification Algorithm

```
In [46]: from sklearn.ensemble import RandomForestClassifier
model_type = RandomForestClassifier(n_estimators=100,criterion="entropy")
outcome_var = ['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
classification_method(model_type, df,predictor_var,outcome_var)
```

```
<ipython-input-44-724901f8b265>:2: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().


Accuracy : 85.232%

<ipython-input-44-724901f8b265>:12: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().

<ipython-input-44-724901f8b265>:12: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().


Cross-Validation Score : 48.059%

<ipython-input-44-724901f8b265>:17: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().
```

In [47]:

```
imp_input = pd.Series(model_type.feature_importances_, index=predictor_var).sort_values(ascending=False)
print(imp_input)
```

```
team2            0.281525
team1            0.217101
toss_winner      0.189192
venue            0.130244
city             0.122602
toss_decision    0.059336
dtype: float64
```

Calculating results using K Neighbors Classification Algorithm

In [48]:
```
from sklearn.neighbors import KNeighborsClassifier
model_type = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
outcome_var = ['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
classification_method(model_type, df,predictor_var,outcome_var)
```

```
Accuracy : 59.574%
Cross-Validation Score : 37.293%

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\neighbors\_classification.py:179: D
ataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\neighbors\_classification.py:179: D
ataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\neighbors\_classification.py:179: D
ataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\neighbors\_classification.py:179: D
ataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
ing ravel().
```

Calculating results using Naive Bayes Classification Algorithm

In [49]:

```
from sklearn.naive_bayes import GaussianNB
model_type = GaussianNB()
outcome_var = ['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
classification_method(model_type, df,predictor_var,outcome_var)
```

```
Accuracy : 22.528%
Cross-Validation Score : 21.391%

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().
```

Calculating results using SVM Algorithm

In [50]:

```
from sklearn.svm import SVC
model_type =SVC(kernel='linear', random_state=0)
outcome_var = ['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
classification_method(model_type, df,predictor_var,outcome_var)
```

```
c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().


Accuracy : 38.924%
Cross-Validation Score : 32.670%

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().

c:\users\keshav\appdata\local\programs\python\python38-32\lib\site-packages\sklearn\utils\validation.py:63: DataConvers
ionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().
```

Calculating results using Decision Tree Classification Algorithm

In [51]:

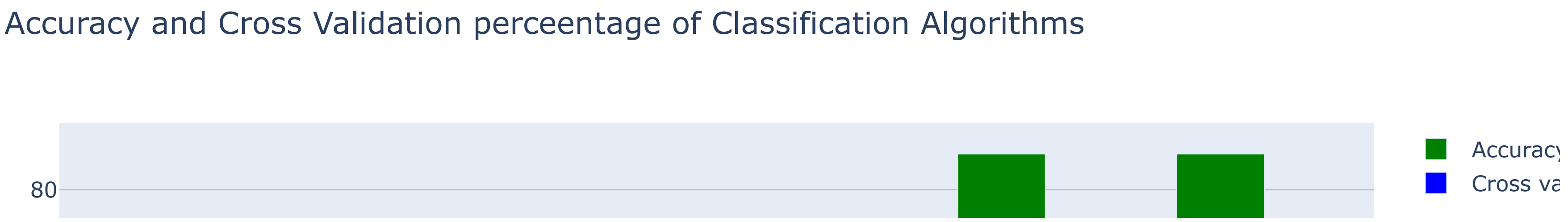
```
from sklearn.tree import DecisionTreeClassifier
model_type = DecisionTreeClassifier(criterion='entropy', random_state=0)
outcome_var = ['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
classification_method(model_type, df,predictor_var,outcome_var)
```

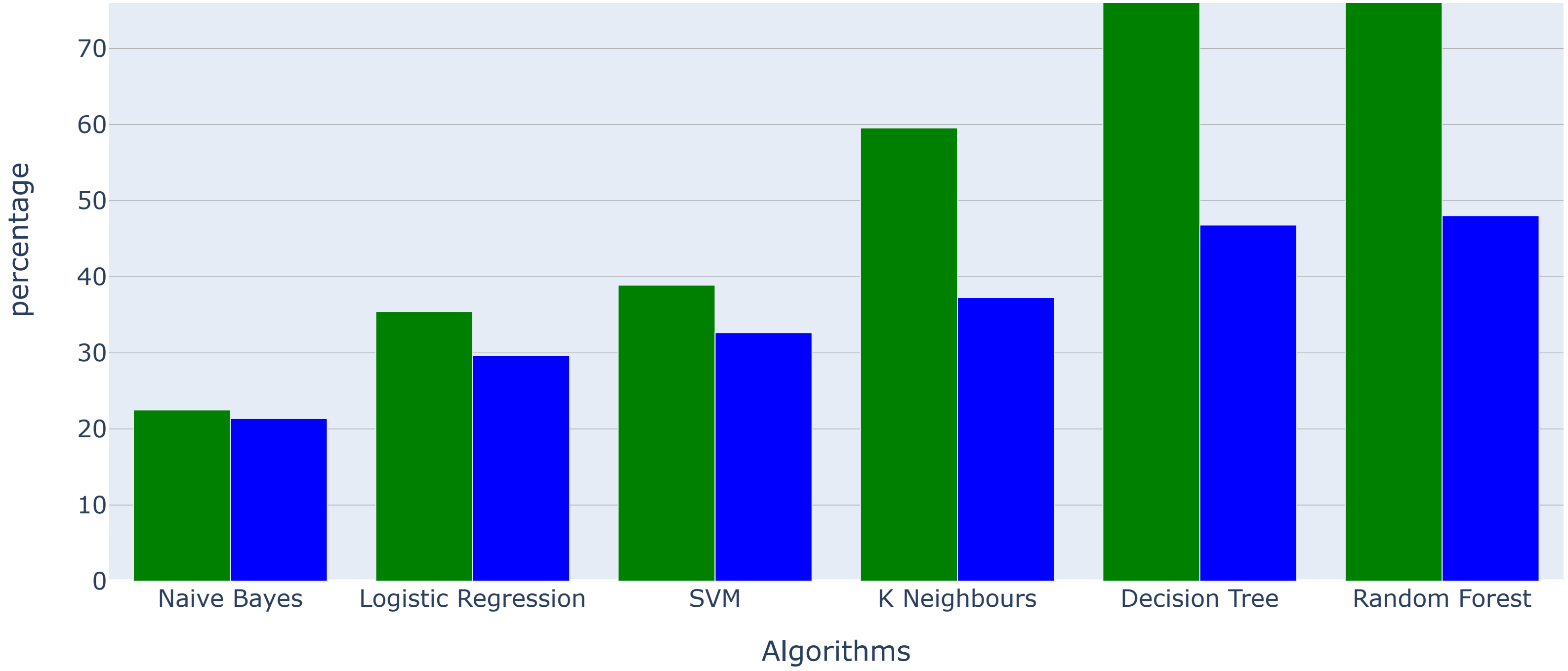
```
Accuracy : 85.232%
Cross-Validation Score : 46.810%
```

In [ ]:

Plotting Accuracy and Cross-Validation percentage for classification algorithms

```
In [53]: import plotly.graph_objects as go
         from plotly.offline import init_notebook_mode, iplot
         labels = [ 'Naive Bayes','Logistic Regression', 'SVM', 'K Neighbours','Decision Tree', 'Random Forest' ]


         y = [22.52, 35.54, 38.92, 59.57, 85.23, 85.23]
         z = [21.39, 29.64, 32.67, 37.29, 46.81, 50.56]
         index = range(1,6)
         df =  pd.DataFrame({'Algorithms' : labels,
                                         'Accuracy' : y,
                                         'Cross validation' : z },
                                         columns=['Algorithms','Accuracy', 'Cross validation'])
         trace1 = go.Bar(x=df['Algorithms'], y=df['Accuracy'],name="Accuracy", marker=dict(color='green'))
         trace2 = go.Bar(x=df['Algorithms'], y=df['Cross validation'],name="Cross validation", marker=dict(color='blue'))

         # Fill out  data with our traces
         data = [trace1, trace2]
         # Create layout and specify title, legend and so on


         layout = go.Layout(title="Accuracy and Cross Validation perceentage of Classification Algorithms",
                            xaxis=dict(title="Algorithms"),
                            yaxis=dict(title="percentage"),
                            barmode="group")


         # Create figure with all prepared data for plot
         fig = go.Figure(data=data, layout=layout)
         # Create a plot in your Python script directory with name "bar-chart.html"
         iplot(fig)
         df
```





Out[53]:

	Algorithms	Accuracy	Cross validation
0	Naive Bayes	22.52	21.39
1	Logistic Regression	35.44	29.64
2	SVM	38.92	32.67
3	K Neighbours	59.57	37.29
4	Decision Tree	85.23	46.81
5	Random Forest	85.23	48.05

Hence, Random Forest and Decision Tree provides the highest accuracy whereas, Naive Bayes provides the least