Name: Vinit Girkar
Student ID: 801194118

IPL Prediction using Machine Learning Algorithms

Abstract:

Cricket is now a game of enchantment, refreshing and physical strength of 22 players. As the popularity of Indian Premier League games increases, it is essential to understand the game results' potential predictors. A cricket match depends upon various factors, and in this work, the factors with significantly influence the outcome of an IPL matches are identified. The advantage at home ground, coin-toss result, decision on first batting or fielding first, and day-to-day games are such popular cricket literature variables. Three machine learning models were trained and used for predicting the outcome of each IPL match.

Introduction:

With Technology growing more and more advanced in the last few years, and in- depth acquisition of data has become relatively easy. As a result, Machine Learning is becoming quite a trend in sports analytics because availability of live as well as historical data. Decision making may be anything including which player to buy during an auction, which player to set on the field for tomorrows match, or something more strategic task like, building the tactics for forthcoming matches based on players previous performance. Player performances decides the win factor of a team. Player performances matters a lot as every team depends on their player to perform good and perform according to match situation. In selecting the line-up for the team, the player performance is taken as a major factor. Batsman performances in recent matches tells about their form, ability to score runs with a healthy strike rate which is a need of twenty-twenty cricket nowadays. Pitch Conditions are very important in cricket game. There are several kinds of pitches on which cricket has been played. Every ground and his own pitch conditions known for bowling pitches or batting pitches. A match's outcome can also be affected by bad weather. Weather conditions also plays a role in deciding results of a match. Players having good batting averages and consistent performances in the recent matches are the ones on which teams rely on. Because they can play a major role in posting a good target score and in chasing, by handling pressure situations.

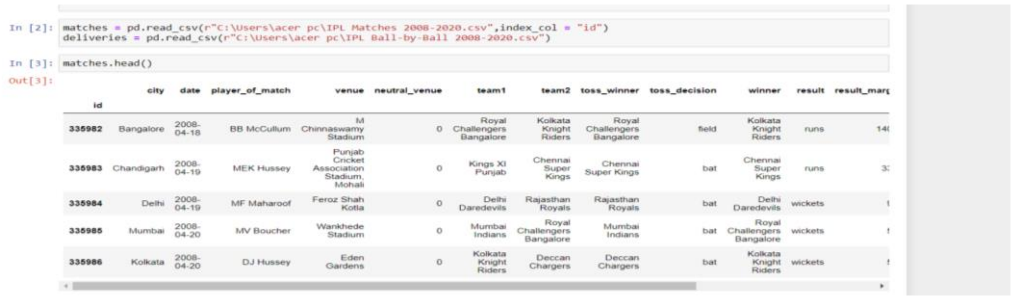
Some interesting machine learning works have also been performed on data acquired from IPL. In a study where it made use of Logistic Regression, Random Forest and Support Vector Machine to predict the outcome of the IPL matches. Some of the interesting work has been done earlier with English County cricket matches where they made use of Naïve Bayes, Logistic Regression, Random Forests, Gradient Boosting algorithm to predict the outcome of English County Cricket Matches. The study was concluded with Naïve Bayes outperforming all other algorithms with the first model giving out average prediction accuracy of 62.4% and second model giving average prediction accuracy of 64%.

Name: Vinit Girkar
Student ID: 801194118

<u>Proposed Work:</u>

Dataset:

The dataset was taken from Kaggle. The data was scraped from the site and maintained in a Comma Separated Values (CSV) format. The dataset contains 2 file which are matches.csv and Ball by Ball (deliveries) bowled during 2008-2020 seasons. The dataset had many features including date, season, home team, away team, toss, winner, man of the match, venue etc. For example, one KKR plays with CSK in its home stadium (Eden Gardens) next time they play against CSK in their home Stadium. So, while make the dataset concept of home and away team are considered to prevent the redundancy. IPL has been 12 years old which is why 689 matches data were available after the pre-processing. Due to certain difficulties with some ongoing team franchises, in some of the seasons the league has seen the participation of new teams and some teams discontinued.



```
In [2]: matches = pd.read_csv(r"C:\Users\acer pc\IPL Matches 2008-2020.csv",index_col = "id")
        deliveries = pd.read_csv(r"C:\Users\acer pc\IPL Ball-by-Ball 2008-2020.csv")

In [3]: matches.head()
Out[3]:
```

id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_marg
335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	Kolkata Knight Riders	runs	14
335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	0	Kings XI Punjab	Chennai Super Kings	Chennai Super Kings	bat	Chennai Super Kings	runs	3
335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	Delhi Daredevils	Rajasthan Royals	Rajasthan Royals	bat	Delhi Daredevils	wickets	[illegible]
335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mumbai Indians	Royal Challengers Bangalore	Mumbai Indians	bat	Royal Challengers Bangalore	wickets	[illegible]
335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight Riders	Deccan Chargers	Deccan Chargers	bat	Kolkata Knight Riders	wickets	[illegible]

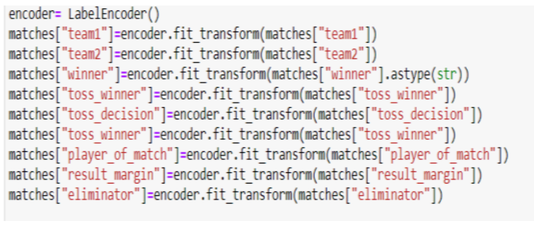
Data Cleaning:

Here we will delete rows with null values and edit duplicate names. Here we notice that Rising Pune Supergiant and Rising Pune Supergiant represent the same team. So, we need to edit it.

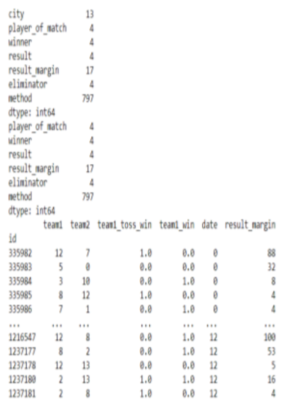
```
matches["team2"]=matches["team2"].replace("Rising Pune Supergiant","Rising Pune Supergiants")
matches["team1"]=matches["team1"].replace("Rising Pune Supergiant","Rising Pune Supergiants")
matches["winner"]=matches["winner"].replace("Rising Pune Supergiant","Rising Pune Supergiants")
matches["toss_winner"]=matches["toss_winner"].replace("Rising Pune Supergiant","Rising Pune Supergiants")
matches["date"] = pd.DatetimeIndex(matches['date']).year
```

Changing Text in Features into Numerical Values:

For the columns containing text, they will only be able to assist the model in the prediction only if they have numerical values. So, we need to change the string values to numerical values.



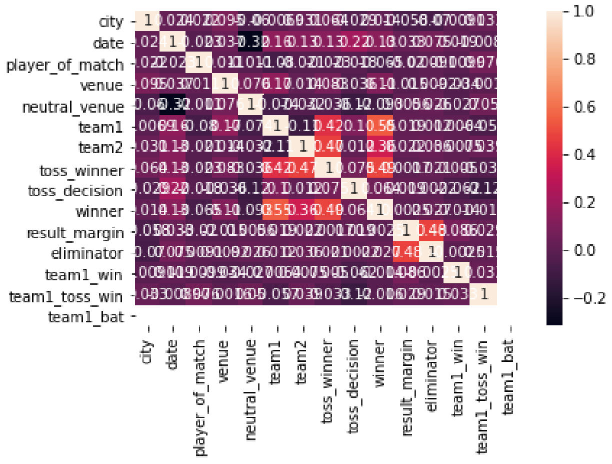
```
encoder= LabelEncoder()
matches["team1"]=encoder.fit_transform(matches["team1"])
matches["team2"]=encoder.fit_transform(matches["team2"])
matches["winner"]=encoder.fit_transform(matches["winner"].astype(str))
matches["toss_winner"]=encoder.fit_transform(matches["toss_winner"])
matches["toss_decision"]=encoder.fit_transform(matches["toss_decision"])
matches["toss_winner"]=encoder.fit_transform(matches["toss_winner"])
matches["player_of_match"]=encoder.fit_transform(matches["player_of_match"])
matches["result_margin"]=encoder.fit_transform(matches["result_margin"])
matches["eliminator"]=encoder.fit_transform(matches["eliminator"])
```



```
city               13
player_of_match     4
winner              4
result              4
result_margin      17
eliminator          4
method            797
dtype: int64
player_of_match     4
winner              4
result              4
result_margin      17
eliminator          4
method            797
dtype: int64
         team1  team2  team1_toss_win  team1_win  date  result_margin
id
335982      12      7             1.0        0.0     0             88
335983       5      0             0.0        0.0     0             32
335984       3     10             0.0        1.0     0              8
335985       8     12             1.0        0.0     0              4
335986       7      1             0.0        1.0     0              4
...        ...    ...             ...        ...   ...            ...
1216547     12      8             0.0        1.0    12            100
1237177      8      2             0.0        1.0    12             53
1237178     12     13             0.0        0.0    12              5
1237180      2     13             1.0        1.0    12             16
1237181      2      8             1.0        0.0    12              4
```

Name: Vinit Girkar
Student ID: 801194118

Make Correlation Matrix:

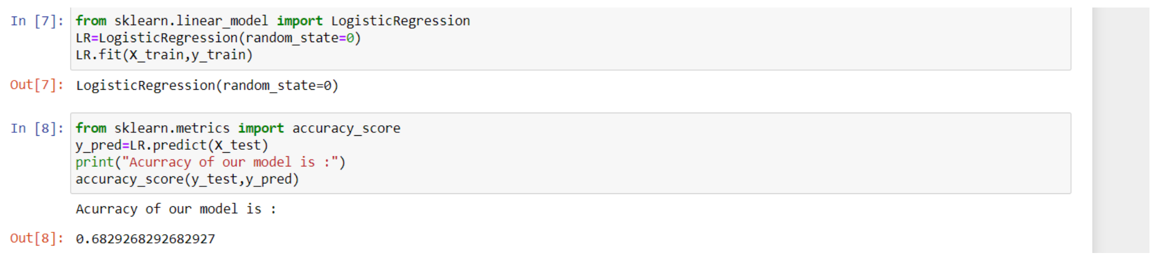


Machine Learning Algorithms:

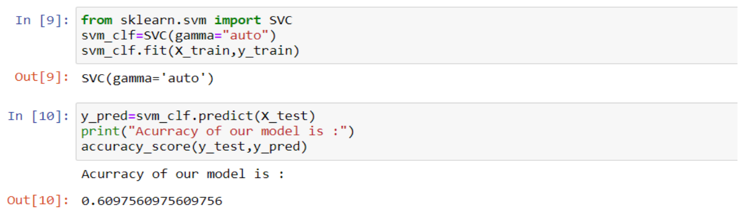
Logistic Regression: A brief description of the model is given below. Suppose $p=Pr(y=1)$ is the probability that an individual has diagnosed asthma, and thus $(1-p)$ represents the probability of not diagnosed. Now, pi is non-linearly related to the explanatory variables as $pi=\exp(X\beta)1+\exp(X\beta)$

Where pi ranges between 0 to1 and X represent explanatory variables, and β refers to the regression coefficients. The odds ratio is $pi1-pi=\exp(X\beta)$

Taking natural logarithm into both sides of the equation would provide the equation for the logistic regression technique, and which is as follows: $ln⍰pi1-pi⍰=X\beta$

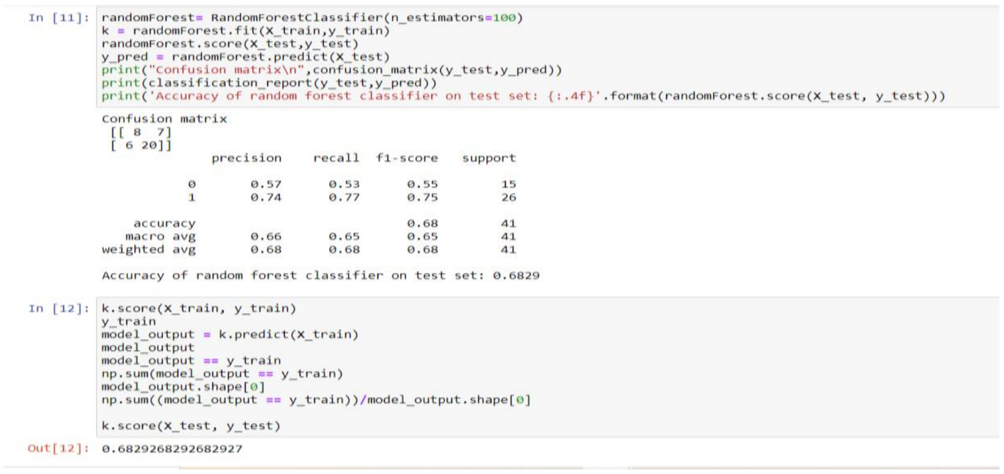

```
In [7]:  from sklearn.linear_model import LogisticRegression
          LR=LogisticRegression(random_state=0)
          LR.fit(X_train,y_train)

Out[7]:  LogisticRegression(random_state=0)

In [8]:  from sklearn.metrics import accuracy_score
          y_pred=LR.predict(X_test)
          print("Acurracy of our model is :")
          accuracy_score(y_test,y_pred)

          Acurracy of our model is :

Out[8]:  0.6829268292682927
```

Support Vector Machine: Support Vector Machine classifier is for classification and regression problems. It is an supervised learning algorithm which itself gets trained from the given past data. In the support vector system hyper –plane is used where the points are placed in that particular area for classification. The hyper plane differentiates the two classes very well and hence classification is done in the n dimensional space. This support vector machine is mainly used for the classification purpose and it gives the best results.


```
In [9]:  from sklearn.svm import SVC
          svm_clf=SVC(gamma="auto")
          svm_clf.fit(X_train,y_train)

Out[9]:  SVC(gamma='auto')

In [10]: y_pred=svm_clf.predict(X_test)
          print("Acurracy of our model is :")
          accuracy_score(y_test,y_pred)

          Acurracy of our model is :

Out[10]: 0.6097560975609756
```

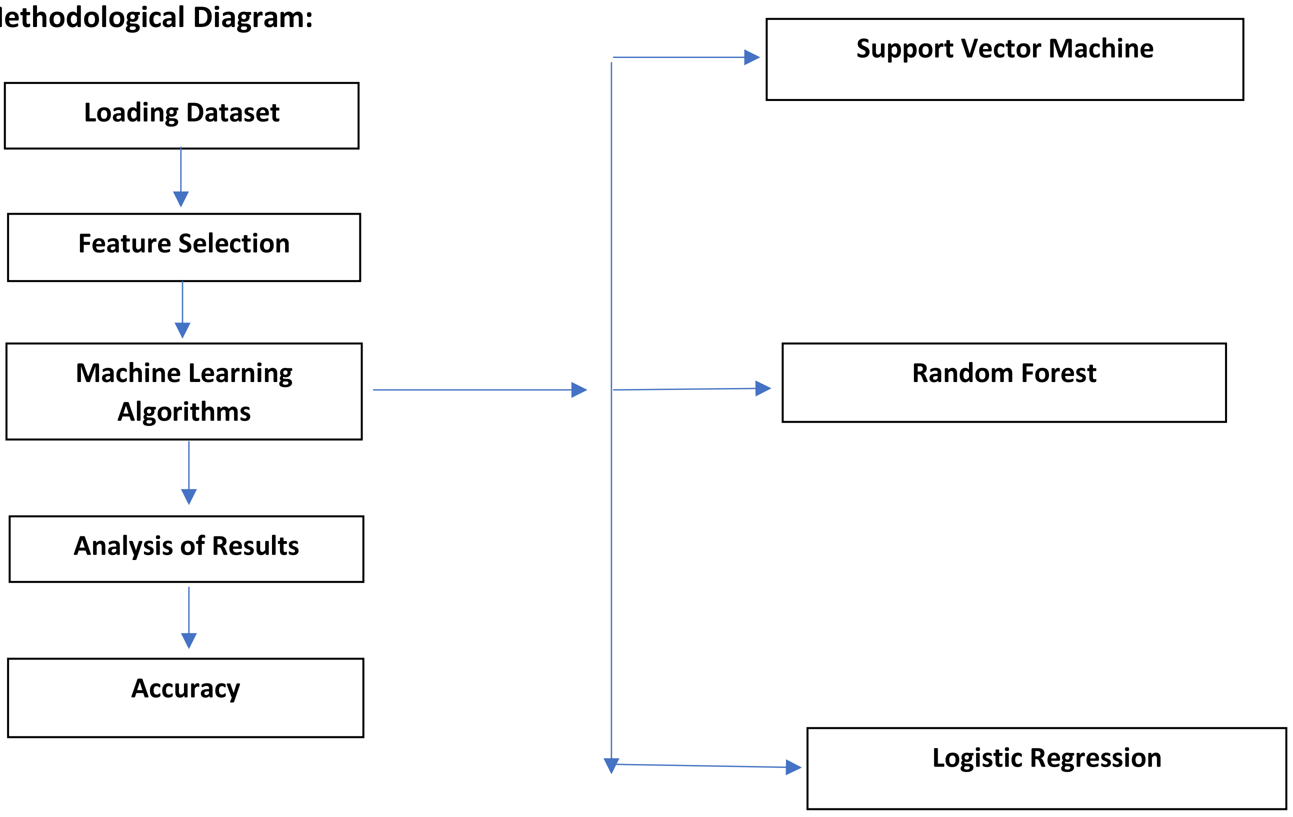
Name: Vinit Girkar
Student ID: 801194118

Random Forest Algorithm: Random Forest classifier is used for classification and regression which is an supervised learning algorithm where the machine learns from the past data and predict the results. Random forest work with the decision trees on data samples and finally gets the best solution among them. It is mainly used for classification problems. Here in this project the random forest has given the best accuracy for the variables that have been taken. The problem of over fitting the data can be done by the random forest effectively.



```
In [11]: randomForest= RandomForestClassifier(n_estimators=100)
         k = randomForest.fit(X_train,y_train)
         randomForest.score(X_test,y_test)
         y_pred = randomForest.predict(X_test)
         print("Confusion matrix\n",confusion_matrix(y_test,y_pred))
         print(classification_report(y_test,y_pred))
         print('Accuracy of random forest classifier on test set: {:.4f}'.format(randomForest.score(X_test, y_test)))

         Confusion matrix
          [[ 8  7]
          [ 6 20]]
                       precision    recall  f1-score   support

                    0       0.57      0.53      0.55        15
                    1       0.74      0.77      0.75        26

             accuracy                           0.68        41
            macro avg       0.66      0.65      0.65        41
         weighted avg       0.68      0.68      0.68        41

         Accuracy of random forest classifier on test set: 0.6829

In [12]: k.score(X_train, y_train)
         y_train
         model_output = k.predict(X_train)
         model_output
         model_output == y_train
         np.sum(model_output == y_train)
         model_output.shape[0]
         np.sum((model_output == y_train))/model_output.shape[0]

         k.score(X_test, y_test)

Out[12]: 0.6829268292682927
```

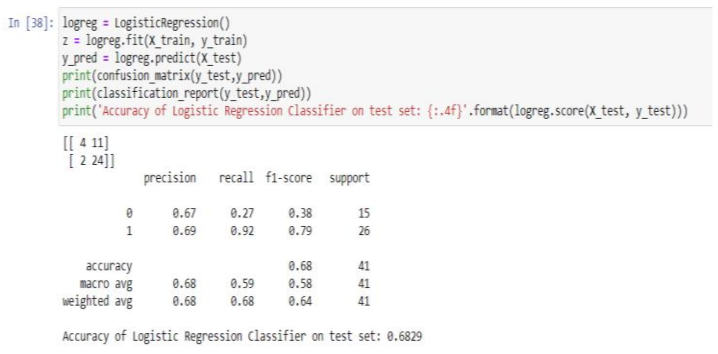
Methodological Diagram:



Name: Vinit Girkar
Student ID: 801194118

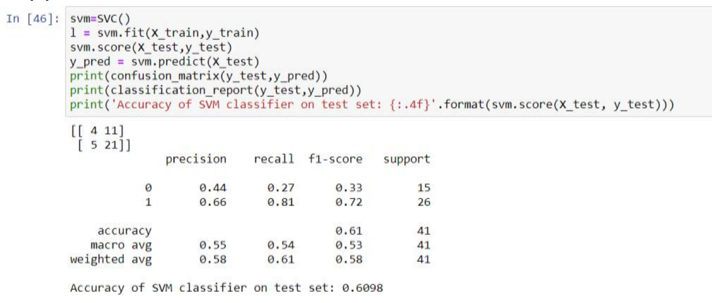
Evaluation:

1. Logistic Regression



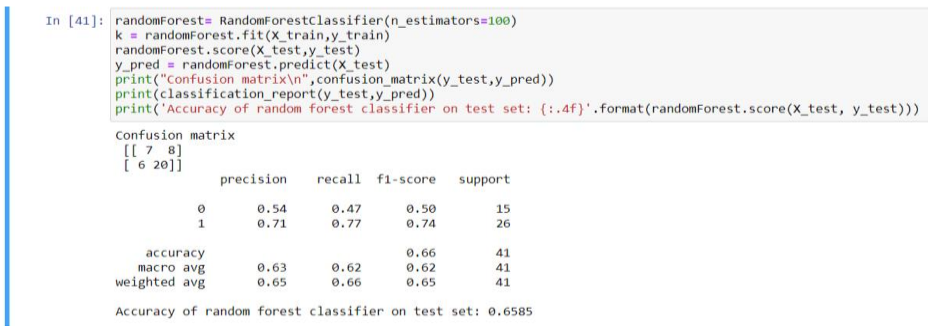
```
In [38]: logreg = LogisticRegression()
         z = logreg.fit(X_train, y_train)
         y_pred = logreg.predict(X_test)
         print(confusion_matrix(y_test,y_pred))
         print(classification_report(y_test,y_pred))
         print('Accuracy of Logistic Regression Classifier on test set: {:.4f}'.format(logreg.score(X_test, y_test)))

         [[ 4 11]
          [ 2 24]]
                       precision    recall  f1-score   support

                    0       0.67      0.27      0.38        15
                    1       0.69      0.92      0.79        26

             accuracy                           0.68        41
            macro avg       0.68      0.59      0.58        41
         weighted avg       0.68      0.68      0.64        41

         Accuracy of Logistic Regression Classifier on test set: 0.6829
```

2. Support Vector Machine:



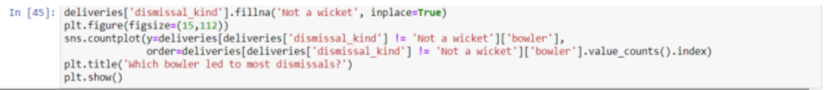
```
In [46]: svm=SVC()
         l = svm.fit(X_train,y_train)
         svm.score(X_test,y_test)
         y_pred = svm.predict(X_test)
         print(confusion_matrix(y_test,y_pred))
         print(classification_report(y_test,y_pred))
         print('Accuracy of SVM classifier on test set: {:.4f}'.format(svm.score(X_test, y_test)))

         [[ 4 11]
          [ 5 21]]
                       precision    recall  f1-score   support

                    0       0.44      0.27      0.33        15
                    1       0.66      0.81      0.72        26

             accuracy                           0.61        41
            macro avg       0.55      0.54      0.53        41
         weighted avg       0.58      0.61      0.58        41

         Accuracy of SVM classifier on test set: 0.6098
```

3. Random Forest Classifier:

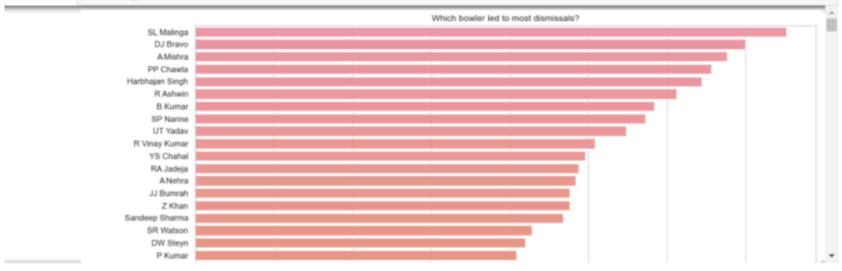


```
In [41]: randomForest= RandomForestClassifier(n_estimators=100)
         k = randomForest.fit(X_train,y_train)
         randomForest.score(X_test,y_test)
         y_pred = randomForest.predict(X_test)
         print("Confusion matrix\n",confusion_matrix(y_test,y_pred))
         print(classification_report(y_test,y_pred))
         print('Accuracy of random forest classifier on test set: {:.4f}'.format(randomForest.score(X_test, y_test)))

         Confusion matrix
          [[ 7  8]
          [ 6 20]]
                       precision    recall  f1-score   support

                    0       0.54      0.47      0.50        15
                    1       0.71      0.77      0.74        26

             accuracy                           0.66        41
            macro avg       0.63      0.62      0.62        41
         weighted avg       0.65      0.66      0.65        41

         Accuracy of random forest classifier on test set: 0.6585
```

4. Dismissal



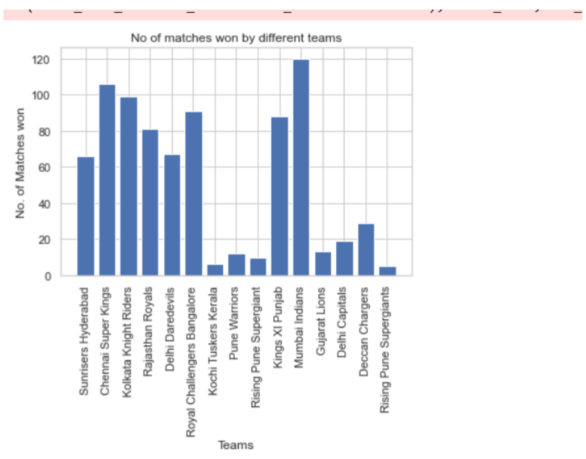
```
In [45]: deliveries['dismissal_kind'].fillna('Not a wicket', inplace=True)
         plt.figure(figsize=(15,112))
         sns.countplot(y=deliveries[deliveries['dismissal_kind'] != 'Not a wicket']['bowler'],
                       order=deliveries[deliveries['dismissal_kind'] != 'Not a wicket']['bowler'].value_counts().index)
         plt.title('Which bowler led to most dismissals?')
         plt.show()
```



```
In [ ]:
```

Name: Vinit Girkar
Student ID: 801194118

5. Matches Won



Conclusion:

The victory of the game mainly depends on the selection of the team and the player performances in the team. Not only the performance but also depends on the some other factors like toss wining, toss decision, venue of the match and lot more. Predicting the IPL is not so easy because the game depends on so many factors. The main source of this paper is that the predicting the winner according to the past data. In this paper three types of classification algorithms were used and predict the results. The tools that are used in implementation are python programming. Among the three algorithms, Logistic Regression gave the highest accuracy of 68.29% than random forest gave the accuracy of 65.85% and support vector system gave the accuracy of 60.98%. This information will be used in the future prediction of winner and team selections accordingly.

References:

1. De Silva, B.M., Swartz, T.B., "Winning the coin toss and the home team advantage in one-day international cricket matches," Department of Statistics and Operations Research, Royal Melbourne Institute of Technology; 1998 Mar
2. Bandulasiri, A., "Predicting the winner in one day international cricket," Journal of Mathematical Sciences & Mathematics Education, 3(1), 6-17, 2008.
3. http://www.ijstr.org/final-print/sep2019/The-Cricket-Winner-Prediction-With-Application-Of-Machine-Learning-And-Data-Analytics-.pdf