**SYBIL - the NFL play predictor**

*Final Report*

*Viraj P Modak*

*December 25 2019*

**EXECUTIVE SUMMARY**

The purpose of this project is to develop a model which can predict whether an offensive play in an NFL game will be a pass play or a run play. The final product, named Sybil, will be of interest to NFL defensive coordinators and their support staff. Sybil was trained using NFL games' play-by-play data collected over the last ten years. Raw data was cleaned up to be represented as features and response. The features are what represent the game situation and the response is pass or run play. Exploratory data analysis and statistical tests were conducted to identify key trends between feature variables and the response. Multiple Machine Learning algorithms were tried including K-Nearest Neighbors, Logistic Regression, Decision Trees and Random Forests. A Random Forest based model resulted in the best accuracy of all - 76%. Few more insights were drawn on the data by segmenting Sybil's results by down, quarter and team. Overall significance of these results is also discussed.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **1.** | **Problem Statement......................................................................................** | **4** |
| **2.** | **Data to be used............................................................................................** | **4** |
| **3.** | **Problem Solving Approach........................................................................** | **5** |
| **4.** | **Data wrangling/clean-up.............................................................................** | **6** |
| **5.** | **Exploratory Data Analysis..........................................................................** | **7** |
| **6.** | **Statistical Analysis.....................................................................................** | **12** |
| **7.** | **Predictive Modeling....................................................................................** | **15** |
| **8.** | **Discussion & Analysis................................................................................** | **21** |
| **9.** | **Significance.................................................................................................** | **23** |
|  | **Appendix......................................................................................................** | **24** |

1. **Problem Statement:**

Imagine you are the defensive coordinator (DC) of an NFL team, devising a strategy for a must win or a play-off game. Your goal is to disrupt plays of the opposition or minimize their impact. That *is* the success criteria of your job. In this situation, imagine that you have a tool to accurately predict what the opposition play is going to be. Would that not be a huge tactical advantage for you? Sybil - the NFL play predictor is that tool.

In an NFL game, there are multiple factors which can decide the offensive play call. They include the play clock, down, yards to go, yards to goal among others. In obvious situations, an experienced practitioner of the game can predict the play call. However, in situations with a significant degree of uncertainty, a human brain may not be able to process all the information. That is where Sybil's data driven decision making and use of ML techniques will be of value.

The target customers for Sybil would primarily be the NFL coaches, in particular the DCs and their support staff. However Sybil can also be marketed to NFL analysts, and enthusiasts such as Fantasy Football players.

1. **Data to be used:**

To train Sybil, the **"nflscrap-R play-by-play"** data is used which is available for free on GitHub at the following location. Data is recorded for every NFL game (regular and post-season) from 2009 through current. Analysis will be restricted to the end of 2018-2019 season.

<https://github.com/ryurko/nflscrapR-data/tree/master/games_data/regular_season>

1. **Problem solving approach:**

Sybil is a model which can predict a Run/Pass play - essentially a binary classification problem. It can be turned into a multi-class classification exercise by adding plays like short/long/right/left pass, QB scramble and right/left run and so on. However, it would be appropriate to build such an enhanced model only ***after*** establishing a proof-of-concept for a much simpler pass/play predictor, which essentially is this project.

Sybil's prediction of pass/run play will be driven by the game situation, which can be summarized using multiple variables (or features) including:

1. Down
2. Yards to go
3. Field position (Yards to goal)
4. Play clock
5. Score line
6. Play formation
7. Team pass performance in the game
8. Team run performance in the game
9. Dual-threat rating of the quarterback (QB)
10. Timeouts remaining

The data clean-up process is described in Section 4. Data was cleaned up to a format which lists out the individual variables and the final outcome. Some of the factors e.g. QB dual threat rating were calculated independently. The Exploratory Data Analysis (EDA) and Statistical Analysis results are reported in Section 5 and Section 6 respectively.

The cleaned data was used to train Sybil using one or more ML techniques. Sybil's performance was evaluated using standard ML performance metrics. The final list of features and the code for training various models is provided in the two Jupyter notebooks listed in the Appendix.

1. **Data Wrangling/Clean-up**

The data wrangling approach and techniques used to prep the data is described in the following steps. A link to a more detailed version of this section is provided in the Appendix.

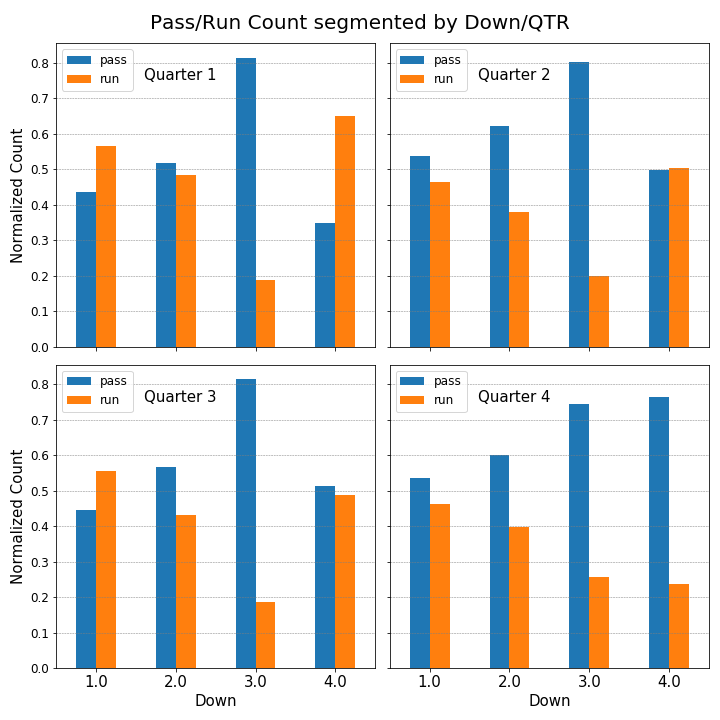
* 1. CSV files for player roster and play by play data were read into separate dataframes.
  2. The play-by-play dataframe was then cleaned up to remove non-play entries such as Timeouts, Two-minute warnings, End of game/QTR, suspensions, resumptions and entries with missing time-stamps. Additional features such as play clock, run/pass performance and the dual threat rating were calculated using individual functions, described as follows:
  3. The game play\_clock was calculated from the game quarter and quarter play clock. It was represented as net minutes remaining in the game going from 60 to 0.
  4. The previous pass performance was represented by (1) historical pass completion percentage for the game and (2) historical pass yardage for the game. Previous run performance was represented by historical run yardage for the game,
  5. QBs which have a proclivity to run the ball are called "dual-threat QBs". Teams with these QBs can add another dimension to their play design. To calculate the dual threat rating of a team, we needed the "dual threat factor" of each quarterback for the team. That was estimated as a product of yardage and run attempts by a QB, normalized by the number of games played. The final dual threat factor was the percentile for all QBs going from 0 through 1.
  6. The dual threat rating calculation for a team in a game proved tricky because of the possibility of multiple QBs playing in the game. The contribution of each QB for a team in the game was "estimated" using the pass count. The team's dual threat rating was then calculated as a weighted average of the QBs' dual threat rating.

This was then used to perform EDA, Statistical Analysis and eventually train Sybil.

1. **Exploratory Data Analysis Results**

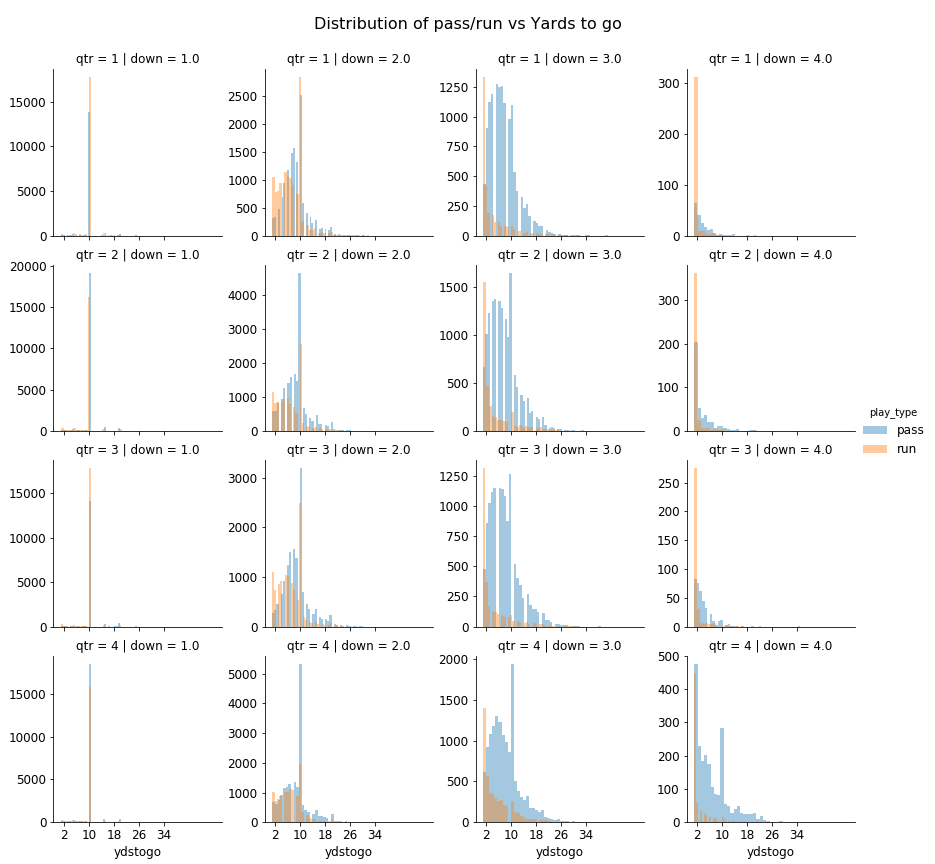
The EDA approach is described in a fully executed notebook and the link for the same is provided in the Appendix. Key highlights and plots are shown here.

1. When segmented by quarters and downs Q1 is when teams are most conservative and prefer to run the ball the most. Q2 and Q4 is when teams are more likely to pass the ball and 3rd down is when the teams are most likely to pass. This is shown in Figure 1.



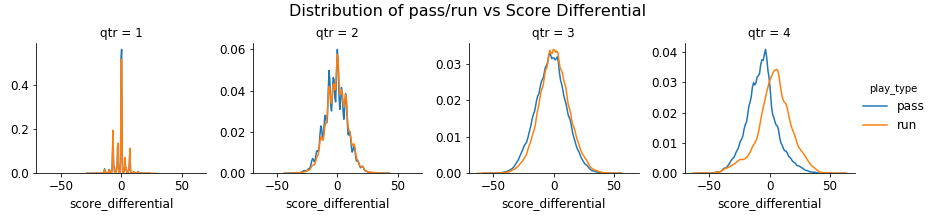
**Figure 1: Count of pass/run plays segmented by play down/qtr**

1. Moving to yardage - in case of long yardage situations as expected, teams prefer to pass on 3rd and 4th downs. Run plays on 3rd and 4th down do happen but only in short yardage situations. On 1st and 2nd downs, visually it is hard to distinguish between pass and run preference. This behavior is shown in Figure 2.



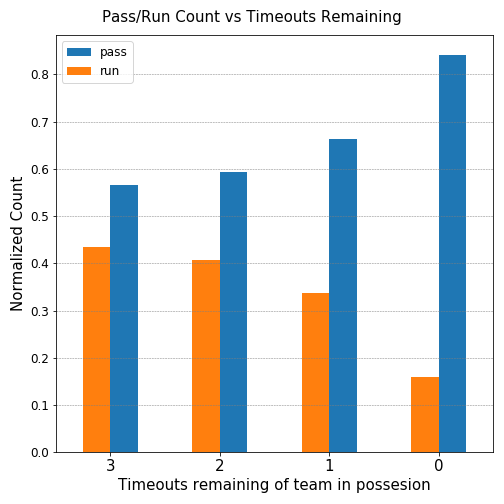
**Figure 2: Count of pass/run plays segmented by play down/qtr and yards to go**

1. Teams are more likely to run the ball if the score difference is positive and more likely to pass if it is negative. This is expected and which is shown in Figure 3.



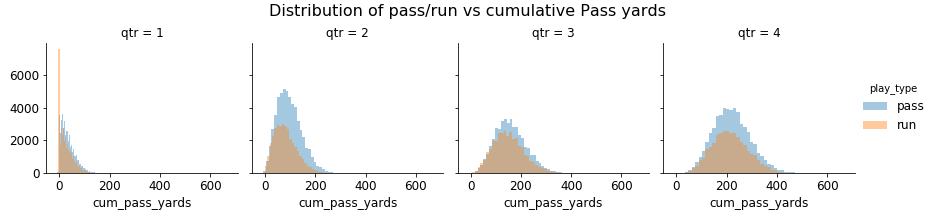
**Figure 3: Count of pass/run plays segmented by play down/qtr and yards to go**

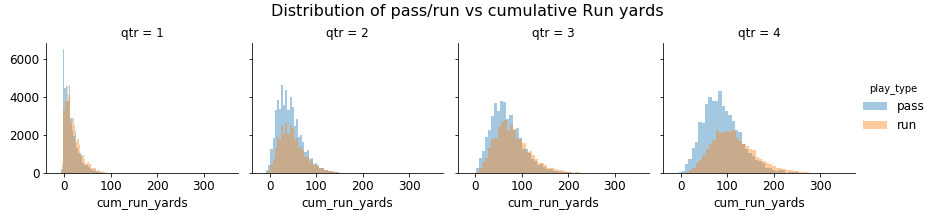
1. In case of timeouts, the tendency of teams to pass increases (and that to run decreases) consistently as they start using timeouts, which is shown in Figure 4



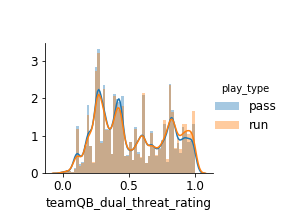
**Figure 4: Count of pass/run plays segmented by play timeouts of team in possession**

1. No strong visual correlation was seen between pass vs run with respect to distance from goal. However, 4th down plays happen if the spot is within specific areas of the field. And there is a very slight preference to pass on 4th down, in Q2 and Q4, if you are further away from goal. The graphic for this is not shown here but is available in the Jupyter notebook.
2. The effect of previous pass/run performance in the game was also tested. This is not a feature directly available in the raw dataset but had to be calculated independently. It will be seen in Section 6, that the effect, although small, is statistically significant. The relevant data is shown in Figure 5.





**Figure 5: Count of pass/run plays segmented by quarter and previous pass/run performance**

1. Another feature which was built and calculated independently, was the dual threat rating of the Team QB. Once again, the difference is not obvious on visual inspection - as seen in Figure 6, but is statistically significant as will be shown in Section 6.
2. 

**Figure 6: Distribution of pass/run plays segmented by team dual threat rating**

1. **Statistical Analysis**

Statistical analysis was performed on continuous variables such as score differential, yards to go or distance to goal, along with hypothesis testing. The results from these report will support the conclusions drawn from the EDA in Section 5. In multiple cases it will be seen that the distributions of the variables are not normal. However, according to the central limit theorem, statistical tests and parameters designed for normal distributions can still be used.

**6.1 Yards to go**

Based on the plot shown in Figure 2, it is worthwhile to look at quarter 3 and possibly quarter 4. Given the distribution a t-test would be appropriate. The null hypothesis in this case is that the compared means are equal. The key statistics, p-values and the statistical inferences are shown in Table 1. To calculate the p-values we assumed that the variances were unequal.

**Table 1: Statistical analysis of yards to go vs qtr/down by play type**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **qtr** | **down** | **pass\_mean** | **run\_mean** | **t-test\_pvalue** |
| 3 | 1 | 10.284213 | 9.828213 | 6.03E-83 |
| 3 | 2 | 8.936741 | 6.735528 | 0.00E+00 |
| 3 | 3 | 7.817511 | 4.561672 | 3.39E-180 |
| 3 | 4 | 3.868852 | 1.829971 | 1.25E-18 |
| 4 | 1 | 10.087728 | 9.72813 | 1.11E-58 |
| 4 | 2 | 8.674929 | 6.91863 | 2.56E-275 |
| 4 | 3 | 7.90534 | 5.397723 | 9.90E-186 |
| 4 | 4 | 7.016202 | 2.767442 | 7.01E-104 |

All p-values are very low, which means that indeed, the difference between yards to go for pass and run is statistically significant. A similar analysis was done for distance from goal, but as consistent trends/differences were not noted.

**6.2 Score differential**

To perform a statistical analysis on score differential, data from only quarters 2, 3 and 4 because the distributions resemble those of a continuous variable. The null hypothesis is that compared means are equal. The key statistics, p-values and the statistical inferences are shown in Table 2 and the variances are assumed to be unequal.

**Table 2: Statistical analysis of score differential vs qtr by play type**

|  |  |  |  |
| --- | --- | --- | --- |
| **qtr** | **pass\_mean** | **run\_mean** | **t-test\_pvalue** |
| 2 | -1.566904 | -0.583737 | 2.98E-55 |
| 3 | -2.27387 | 0.37842 | 7.41E-189 |
| 4 | -6.564699 | 3.633725 | 0.00E+00 |

The numbers are interesting because, to the human eye it is near impossible to spot a difference between the two means except for quarter 4. However, even for qtr 2 and 3 we do see that the differences between means is statistically significant. It is also important from a practical standpoint because the p-value approaches zero with each passing quarter and during qtr 4, the difference is beyond doubt as the p-value is zero. Play quarter (which is used here to segment the data) is a discrete quantity. The same behavior is also studied but segmented by binned play clock, which is a continuous variable. That analysis makes a more impactful visual in a video format. It is included as a link in the appendix as well as in the powerpoint slides.

**6.3 Cumulative Run/Pass performance:**

The next factor studied is teams' historical pass performance and run performance. The hypothesis in this case can is a little tricky to frame but can be expressed as follows: difference between pass and run performance for pass plays is greater than difference between pass and run performance for run plays. It can be represented as follows:

*µp > µr*

where,



The results for the 1-sided t-test are listed in Table 3. In this case the p-value is 1, which means that the likelihood of null hypothesis being true is certain.

**Table 3: Comparing difference between pass and run performance for pass and run plays**

|  |  |  |  |
| --- | --- | --- | --- |
| **diff\_for\_pass\_play** | **diff\_for\_run\_play** | **t-test\_statistic** | **t-test\_pvalue** |
| 75.859833 | 52.317236 | 87.260388 | 1 |

**6.4 Dual threat rating:**

The hypothesis in this case will be that the teams with higher dual threat rating prefer to run the ball more than teams with lower dual threat rating. It can be framed as follows:

*µrun > µpass*

*µpass* = mean dual threat rating for all pass plays

*µrun* = mean dual threat rating for all run plays

The results shown in Table 4 show that the difference in the means is really subtle at face value - also seen in Figure 6 - but statistically significant with the large sample size since the p-value is 1.

**Table 4: Compare dual threat rating for pass plays and run plays**

|  |  |  |
| --- | --- | --- |
| **dual\_threat\_mean\_run** | **dual\_threat\_mean\_pass** | **t-test\_pvalue** |
| 0.538057 | 0.520613 | 1 |

In summary we can say that, just by doing exploratory data analysis, it is possible to extract meaningful trends from subtle differences. Moreover, even though the differences seem subtle to the human eye, given the large sample size, these differences are indeed statistically significant.

1. **Predictive Modeling**

In this section, results from the various ML techniques which were used to train our data are presented. These include\*:

1. K-Nearest Neighbors (KNN)
2. Logistic Regression (LogReg)
3. Decision Trees (DT)
4. Random Forests (RF)

*\*SVMs were tried but ultimately not included in the report as they were noticeably slower compared to all other models and did not perform better than any of the other techniques*

These are standard ML techniques and no further explanation is provided on how the techniques work. With each technique, a preliminary training model was built using "as is" features. Standard performance metrics such as net accuracy score, precision, recall, confusion matrix and the ROC curve were calculated for these preliminary models. The existing features do include engineered features, viz. historical pass/run performance and dual threat rating. However, additional feature engineering was also attempted to improve model performance.

**7.1 Preliminary model training:**

Data is split using 80/20 Train/Test ratio. For each model, hyperparameters are tuned using a grid search over 5-fold cross validation. The final hyperparameters and the performance metrics are presented in Table 5. The ROC curves and confusion matrices are presented in Figure 7 and Figure 8 respectively.The following standard definitions are used to calculate the performance metrics. Defining our classes (pass/run) as either "positive" or "negative" does not make much sense since they can be interchangeable. Hence the definitions come with the caveat that the performance metrics can be calculated for both play types independently. However, calculating the confusion matrices as well as the ROC curves was more straightforward after using numeric notations. In those cases, "1" implies a pass play and 0 implies a run play.

Accuracy =, TP = True positive, TN = True negative

Precision =, FP = False positive

Recall =, FN = False negative

**Table 5: Performance parameters of preliminary model training**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **KNN** | **LogReg** | **DT** | **RF** |
| **Best hyperparameter grid/value** | n\_neighbors = 39 | C = 100 | max\_depth = 13 | max\_depth = 20  min\_samples\_leaf = 4  min\_samples\_split = 2 |
| **Training set accuracy** | 67% | 72% | 76% | 82% |
| **Test set accuracy** | 65% | 72% | 73% | 75% |
| **Precision** | Run: 0.61  Pass: 0.66 | Run: 0.66  Pass: 0.76 | Run: 0.68  Pass: 0.77 | Run: 0.71  Pass: 0.78 |
| **Recall** | Run: 0.44  Pass: 0.80 | Run: 0.67  Pass: 0.76 | Run: 0.68  Pass: 0.77 | Run: 0.70  Pass: 0.79 |

In all cases, expectedly the model performs better on the training set than on the test set. But the difference is reasonable enough that we can assume the model is not over or under-fitting. KNN is the poorest performing model as it relies only on proximity in the feature space. LogReg, DT perform better because they add more complexity to the model; Logreg incorporates a linear dependence on the features and DT relies on a complex hierarchy of if/else questions to reach a final outcome. The weights of the LogReg model are reported in Table 6. Shotgun formation, down and previous pass performance are more influential than others. RF performs even better because of multiple iterations over the training data which incorporates a "wisdom of the crowd" aspect in training.

**Table 6: LogReg coefficient by training feature**

|  |  |
| --- | --- |
| **Feature** | **Coefficient** |
| qtr | -0.123501 |
| down | 0.554158 |
| ydstogo | 0.089554 |
| shotgun | 1.468002 |
| no\_huddle | -0.009586 |
| posteam\_timeouts\_remaining | -0.152588 |
| score\_differential | -0.024096 |
| play\_clock | -0.009307 |
| yards\_to\_goal | 0.001771 |
| cum\_pass\_comp% | 0.551482 |
| cum\_pass\_yards | 0.002881 |
| cum\_run\_yards | -0.007682 |
| teamQB\_dual\_threat\_rating | -0.45137 |



**Figure 7: ROC curves for preliminary model training. Inner textboxes correspond to the particular model. KNN is the worst performer while RF is the best performer**



**Figure 8: Confusion matrices for preliminary model training. Inner textboxes correspond to the particular model. There is a general tendency to predict Pass plays over Run plays. KNN, in fact performs the best on Pass plays but is poorest on Run plays, which impacts its average accuracy score reported in Table 5. For this purpose: 0=Run, 1=Pass**

**7.2 Feature Engineering:**

Additional Feature Engineering was used as follows to improve the performance of the models:

1. Categorical feature for game quarter was represented using "one-hot encoding". There are other categorical features including play down and time-outs remaining were not converted to one-hot encoded features as they are more fluid than game quarter
2. Previous play (pass/run) was added as a feature
3. Continuous variables such as yardage, play\_clock, score\_differential, previous pass/run performance were normalized

Feature Engineering led to only marginal improvement of model performance compared to the preliminary models. This be due to (1) existing features have enough complexity that adding more using one-hot encoding does not enhance the feature space and (2) features don't differ by orders of magnitude which leaves feature scaling redundant. Model hyperparameter values and model accuracy are reported in Table 7. Given that KNN is the worst performer among the models tested, it was excluded from this analysis. ROC curves and confusion matrices are also largely similar and not shown here to avoid redundancy.

**Table 7: Performance metrics for model training post Feature Engineering**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **LogReg** | **DT** | **RF** |
| **Best hyperparameter grid/value** | C = 10 | max\_depth = 11 | max\_depth = 20  min\_samples\_leaf = 4  min\_samples\_split = 2 |
| **Training set accuracy** | 72% | 75% | 82% |
| **Test set accuracy** | 72% | 74% | 76% |

Given that RF with feature engineering gives the highest accuracy seen so far, this will constitute the final product - **Sybil, the NFL play predictor**. Further insights on model accuracy are presented in Section 8 and significance of these results is presented in Section 9.

**7.3 Feature Importance:**

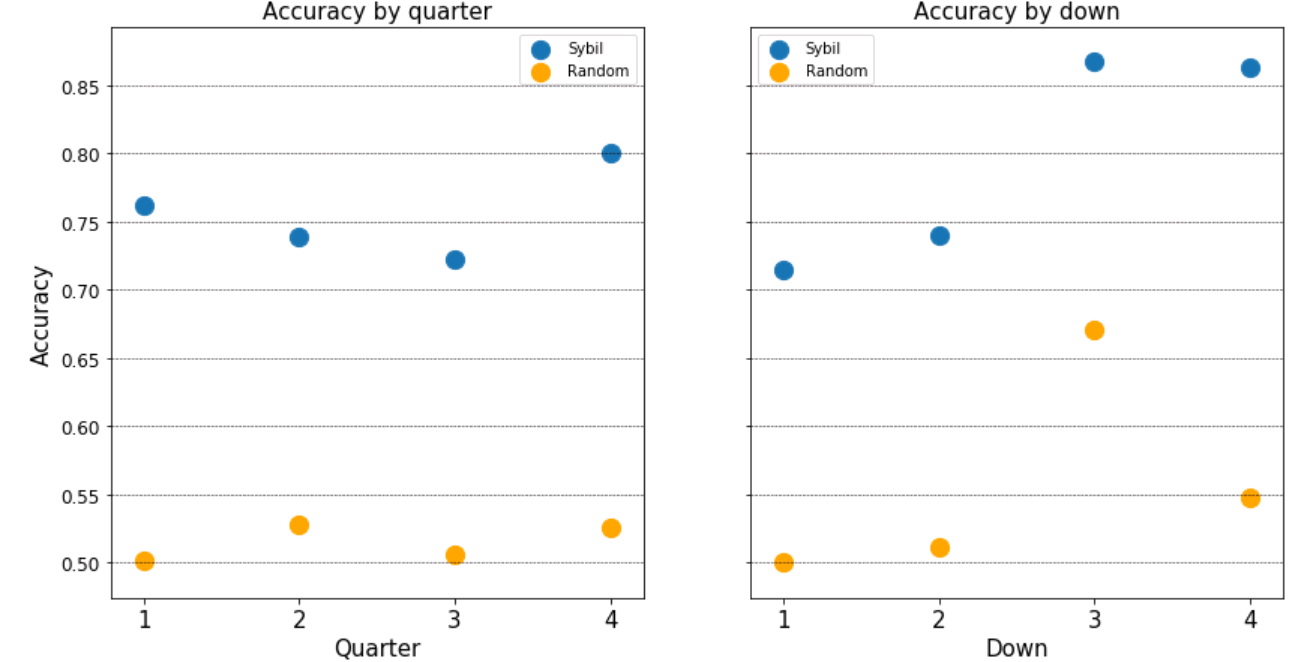
In the previous sections, it was observed that Random Forest was the best performing ML technique, where Section 7.1 pertained to a basic model being trained and Section 7.2, pertained to using additional Feature Engineering. Feature importance can be studied to explore which features are more influential than others. Feature importance for both approaches is listed in Table 8. The "Engineered Features", i.e. the one-hot encoded quarters, previous play result in very low additional importance to the model. However, it should be noted that cum\_run\_yards, cum\_pass\_yards and dual\_threat\_rating are engineered features already and do not exist in the raw data. This might be why the improvement which we observe after additional feature engineering is limited.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Preliminary Model Building** | **Feature Engineering** |
| shotgun | 0.2732 | 0.2629 |
| cum\_run\_yards | 0.103 | 0.1033 |
| play\_clock | 0.0951 | 0.0934 |
| ydstogo | 0.0884 | 0.0812 |
| down | 0.0824 | 0.0801 |
| cum\_pass\_yards | 0.074 | 0.074 |
| score\_differential | 0.0709 | 0.069 |
| teamQB\_dual\_threat\_rating | 0.0614 | 0.06 |
| yards\_to\_goal | 0.0597 | 0.0583 |
| cum\_pass\_comp% | 0.0586 | 0.0583 |
| posteam\_timeouts\_remaining | 0.0157 | 0.0148 |
| prev\_pass\_1.0 |  | 0.0125 |
| prev\_pass\_0.0 |  | 0.0116 |
| qtr\_4 | 0.0137 | 0.0077 |
| qtr\_2 | 0.005 |
| qtr\_3 | 0.0041 |
| no\_huddle | 0.0041 | 0.0038 |

**8. Discussion & Analysis:**

From the confusion matrices it can be seen, the model tends to perform better for pass plays, because pass plays are more frequent. Overall model accuracy is a more robust metric, because it considers correct predictions on both play types. However, is Sybil truly better than a completely "Random" predictor? The accuracy of a "Random" predictor will depend on the pass:run skewness. More information on this behavior is presented in the appendix and an empirical function for this relationship is also reported. For our case, pass:run ratio is 58:42. A truly random predictor for this ratio will be accurate ~51% of the time. A biased predictor which predicts all passes will be 58% accurate. Sybil, with its 76% accuracy is better than both these predictors.

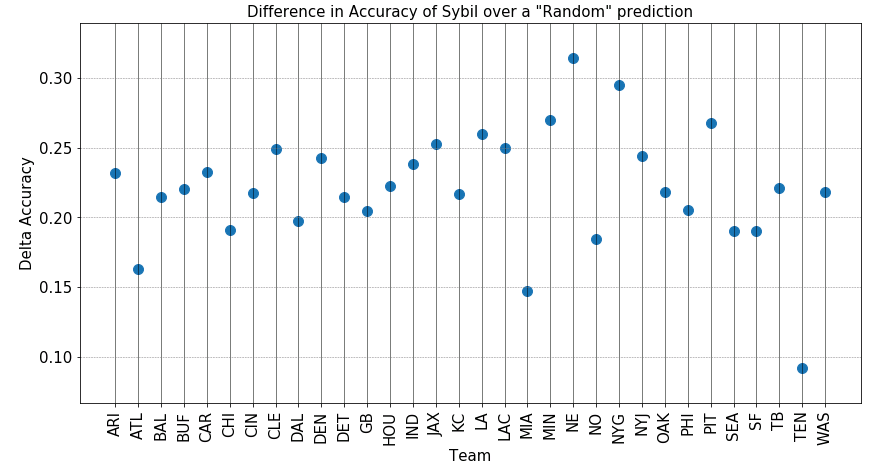
Can we segment out accuracy by various game situations? Let us consider accuracy over qtr and down shown in Figure 9. Accuracy based on a "Random" prediction is between 50-55% for all qtrs and downs except on 3rd down. The difference between the blue and the orange points in the plots essentially show how much better Sybil is compared to a random prediction. We notice that this difference is largest in qtr 4 and on down 4. This can be explained because in these two situations, teams are more likely to play by the situation and avoid taking any risks. This short analysis offers a key insight into where teams might want to add a bit of unpredictability in their play-calling.



**Figure 9: Accuracy score segmented out by qtr and down. Teams are most predictable during qtr 4 and on down 3 and 4**

Sybil's performance over teams can also be investigated. Let's take the case of the latest 2018-2019 regular season. Sybil was implemented for each team individually only for the regular season games. The difference between Sybil's accuracy and that of a "Random" predictor was calculated. This difference is plotted as a scatter - shown in Figure 10. This information was correlated to the regular season record. However, no strong trends were seen.

But going into the play-offs, the predictability metric will be of interest to teams. For example, NE performs more accurately than a "Random" predictor - making NE more predictable than any other team. In spite of that NE qualified for the play-offs. If the oppositions had based their strategy on this model, would they have been able to break NE's plays? Or was NE's execution flawless - and the opposition could not break their plays even after predicting them. These questions will be important as teams devise their strategies going into the play-offs. And this model will provide keen insights as they analyze the performance of their potential opposition.



***Figure 10: Difference between Sybil's accuracy and that of a random predictor over regular season games by Team***

1. **Significance:**

It is generally agreed that predicting the outcome of sporting events is hard.1,2 And predicting a "micro-event" such as a play-call in an NFL game adds another level of complexity to it. Indeed, the game situation does dictate it to some degree. Consider the following situation: 3rd down in the 4th quarter, with 10+ yards to go and the offense is trailing a two-possession game. It will not take a veteran to predict a pass play. But can that prediction be made with 100% certainty? What about a bluff? In sports, there is always this possibility of a "what-if", because one is essentially trying to predict how a human will think and act, and that is incredibly hard.

Moreover, fundamental principles of statistics are often not applicable in sports and are relegated in favor of a more practical perspective. Let's take another example, albeit manufactured. The hypothesis is that A can throw a longer pass than B can. In an experiment allowing 10 throws to each:

A makes the following throws: 50, 52, 53, 51, 48, 46, 48, 47, 40, 35

B makes the following throws: 48, 49, 46, 45, 46, 44, 42, 44, 41, 40

The difference between the two datasets is **NOT** statistically significant. However, from a practical standpoint, there is some truth to the hypothesis. Indeed, if we collect enough data, we might reach a difference which is statistically significant. But will that difference be significant enough to influence a key decision in a game? One cannot answer that with certainty. A running back may average over 100 yards per game going into the super-bowl, but those statistics cannot predict his performance during the game.

Coming back to the problem of NFL play prediction, because it is a binary classification problem, a simple coin flip will be able to make accurate predictions 50% of the time. But because we know how challenging it is to predict an outcome of an event in sport, anything above 50% is significant. Sybil goes well and truly beyond that with an average accuracy of 76% and between 70-85% for specific cases. There is of course room for improvement. For example pictures of the formation prior to the snap can be a key addition to the feature space. This can also help make a finer prediction such as run direction, QB scramble and pass location/depth. Nonetheless, even with its current capability NFL teams are better off having Sybil assist them in predicting plays.

.

**Appendix**

1. References cited in Section 9:
   1. Stekler et al, *Issues in sports forecasting*, International Journal of Forecasting, 2010
   2. Aoki et al, *Luck is Hard to Beat: The Difficulty of Sports Prediction*, Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017
2. Link to the cleaned-up dataset:

<https://drive.google.com/open?id=1sL2ZLX1BU7o800UiPCSJ2n84D3Y5GUGJ>

The raw dataset is not uploaded to a shared location for this project, but can be accessed freely on the link provided in Section 2

1. Link to the Python files for data wrangling/clean-up:

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/Python_Code.zip>

1. Link to the Jupyter Notebook for Exploratory Data Analysis:

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/VirajModak_Capstone1_DataStorytelling.ipynb>

1. Link to the Jupyter Notebook for Statistical Analysis:

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/VirajModak_StatisticalAnalysis_20191128.ipynb>

1. Links to the Jupyter Notebooks for Machine Learning:

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/PredictiveModeling.ipynb>

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/PredictiveModeling_FeatureEngineering.ipynb>

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/PredictiveModeling_FeatureEngineering_withAnalysis_rev2.ipynb>

1. Link to short video of Play type segmentation by play clock and score differential

<https://github.com/virajmodak16/NFL_Play_Predictor/blob/master/PlayType_by_Clock_and_DeltaScore.avi>

**Probability of a Random Predictor:**

Let us consider a distribution with an exact 50% pass/run ratio with 1 denoting a pass and 0 denoting a run. Let us design a random predictor which bases its prediction based on a number drawn from this distribution. If we were to run several simulations of this exercise, and calculate the accuracy of this predictor, on an average it will be accurate 50% of the time.

Now let's consider the extreme case - with a 100% pass/ratio. Now, our random predictor will draw a 1 every single time and as a result, will be accurate 100% of the time. We can calculate how the accuracy of this random predictor trends over a range of pass/run ratios by running simulations in this range. This is shown in script and a trend plot as follows. The resulting function is shown as a plot insert.

1. **def** func(x, a, b, c, d):
2. **return** a\*(x \*\* b) + c\*(x) + d
4. **import** random
5. **from** scipy.optimize **import** curve\_fit
6. **from** operator **import** eq
7. **import** matplotlib.pyplot as plt
8. **import** numpy as np
10. prob\_vals = []
11. range\_vals = np.linspace(50,100,51)
12. **for** j **in** range\_vals:
13. base = ['pass']\*int(100-j)+['run']\*int(j)
14. random.Random(4).shuffle(base)
15. sum\_selection = []
16. **for** i **in** range(10000):
17. selection = []
18. **for** i **in** range(100):
19. selection.append(random.choice(base))
20. sum\_selection.append(sum(list(map(eq, base, selection))))
21. prob\_vals.append(np.mean(sum\_selection))
23. popt, pcov = curve\_fit(func, range\_vals, prob\_vals, maxfev=10000)
24. plt.scatter(range\_vals, prob\_vals)
25. plt.xlabel('Hypothetical Percent of Passes total data')
26. plt.ylabel('Total Accuracy of Random prediction')
27. plt.title('How a random choice model would look like\nif percent of passes in data is varied')
28. plt.plot(range\_vals, func(range\_vals, \*popt), 'r-')#...          label='fit: a=%5.3f, b=%5.3f, c=%5.3f' % tuple(popt))
29. plt.legend(('Fit', 'Simulation'))

