NFL Go For It!

by Mike Ghirardo and Thomas McCann

In football there are many decisions a team needs to make in order win the game. In this project we focus on the decision that needs to be made on the 4th down of any given play. There are three decisions to be made on the fourth down.
1. Punt the ball
2. Kick a field goal
3. Go for a first down

In this project we try to determine which decision should be made under certain conditions. The following are the conditions which we take into account in determining the decision.
1. Offensive and defensive rank of the offensive team.
2. Offensive and defensive rank of the defensive team.
3. The number of yards to convert for a first down.
4. The field position started from.

With this information from the data we were able to estimate the expected points scored for each of the of three decisions.
Finally, with this information a decision can be made.

In [1]:

```
import os
import csv
import math
import numpy as np
import pandas as pd
import statsmodels.formula.api as sm
import matplotlib.pyplot as plt
from sklearn import neighbors
import seaborn as sns
from pylab import *
from mpl_toolkits.mplot3d.axes3d import Axes3D
from matplotlib.backends.backend_pdf import PdfPages
from matplotlib.lines import Line2D
import ipythonblocks
import Image, ImageDraw, ImageFont, ImageOps
import IPython.core.display as icd
%load_ext rmagic
%pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib

WARNING: pylab import has clobbered these variables: ['linalg', 'draw_if_inte
ractive', 'random', 'power', 'info', 'fft']
`%pylab --no-import-all` prevents importing * from pylab and numpy
```

Importing NFL play-by-play data from years 2002 to 2012.

In [2]:

```
os.system('curl -L -o 2002_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMNGQzY2YyNmUtMTlhYy00YmQyLTg3ZTUtMGI2NDhjNGU4Zjg5&export=download"'
)
os.system('curl -L -o 2003_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMODljMmIxNzItNzJjNy00ODJiLWJiNDItMDBlZGMwMjkwOTlk&export=download"'
)
os.system('curl -L -o 2004_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMMGUyNjVkMWEtOWE2YS00YzI3LWJjYjEtZWU2MTIyNmJhOTk0&export=download"'
)
os.system('curl -L -o 2005_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMYmIyMTdjNWItMjhiNS00NjJkLWIyYWEtZmI1ZGM4NmFmZGQy&export=download"'
)
os.system('curl -L -o 2006_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMN2YxNGM0MzUtYTc2Mi00YjVjLWI3N2EtMzIwMDA0Y2E5OTg1&export=download"'
)
os.system('curl -L -o 2007_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMYWZkOWU1YTItYTUzNS00MmM4LTk1MTktYmI3Y2E1Zjc3OTIy&export=download"'
)
os.system('curl -L -o 2008_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMZDJmYzIzNWQtNjIyNS00NzQzLWJiMTEtYWI5M2U0MTI4Njlk&export=download"'
)
os.system('curl -L -o 2009_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMMDAyOGRhMjYtMzlkMC00NGQwLTgxMWUtOWNmYWMxY2Q2ODY3&export=download"'
)
os.system('curl -L -o 2010_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMMWRkMDc0MDgtZDZhMi00ZGRlLTlkYjEtOTNkZjViZDI0ZGY2&export=download"'
)
os.system('curl -L -o 2011_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMbmZvYzE3cjBzblE&export=download"')
os.system('curl -L -o 2012_nfl_pbp_data.csv "https://docs.google.com/uc?id=0B
xEXxf9odCnMMC1sR3dtNEtHLW8&export=download"')
```

Out[2]:

```
0
```

Joining data sets together to get one long dataset of play-by-play data accross all 11 years.

In [3]:

```
pd.set_option('display.line_width', 300)
pbp_data = pd.read_csv('2002_nfl_pbp_data.csv')
yard = range(0,100)
teams = list(set(pbp_data.off))[1:]
seasons = range(2002,2013)
```

```
for i in seasons[1:]:
    pbp_data = pbp_data.append(pd.read_csv('%d_nfl_pbp_data.csv' % i), ignore
_index = True)
```

```
//anaconda/lib/python2.7/site-packages/pandas/core/config.py:570: Deprecation
Warning: line_width has been deprecated, use display.width instead (currently
 both are
identical)

  warnings.warn(d.msg, DeprecationWarning)
```

Taking out post-season games to get a more accurate ranking of teams.

In [4]:

```
pbp_dataTF = pbp_data['gameid'].map(lambda x: str(x).endswith(('01', '02'), 4
, 6)) == False
pbp_dataTF[pbp_data['gameid'].map(lambda x: str(x).endswith(('20050102', '200
60101', '20100103', '20110102', '20120101'), 0, 8)) == True] = True
pbp_dataT = pbp_dataTF[pbp_dataTF == True]
pbp_data = pbp_data.loc[pbp_dataT.index]
pbp_data = pbp_data.reset_index(range(len(pbp_data)))
pbp_data = pbp_data.drop(['index'], 1)
```

The following retrieves the first and last plays of each game. This will be useful in helping us determine team ranks by how many points scored per game and how many points let go per game.

In [5]:

```
shifted_data1 = (pbp_data.shift(1)).shift(-1)
shifted_data2 = pbp_data.shift(1)
shifted_data2.rename(columns=lambda x: 'last' + x, inplace=True)
shifted_data = shifted_data1.join(shifted_data2, how = 'outer')
offTF = shifted_data['off'] == shifted_data['lastoff']
defTF = shifted_data['def'] == shifted_data['lastdef']
gameIDTF = shifted_data['gameid'] == shifted_data['lastgameid']
first_last_play = shifted_data[offTF == False]
first_last_play = shifted_data[gameIDTF == False]
```

The following sums points gained and points let go per game per team per season. The means by which the teams are ranked offensively and defensively is taking the total number of points scored and total number of points let go and adding them.

In [6]:

```
rank_off_off_score = pd.DataFrame(first_last_play.groupby(['lastseason', 'las
toff'])['lastoffscore'].sum())
rank_off_def_score = pd.DataFrame(first_last_play.groupby(['lastseason', 'las
tdef'])['lastdefscore'].sum())
rank_def_def_score = pd.DataFrame(first_last_play.groupby(['lastseason', 'las
toff'])['lastdefscore'].sum())
rank_def_off_score = pd.DataFrame(first_last_play.groupby(['lastseason', 'lastdef'])['lastoffscore'].sum())
team_score_off_rank = rank_off_off_score.join(rank_off_def_score)
team_score_off_rank['total'] = team_score_off_rank['lastoffscore'] + team_score_off_rank['lastdefscore']
team_score_def_rank = rank_def_def_score.join(rank_def_off_score)
team_score_def_rank['total'] = team_score_def_rank['lastoffscore'] + team_score_def_rank['lastdefscore']
```

Creating a two matrices of the total points scored and total points let go with the season as the column and the team as the row, and then ranking them to get the offensive and defensive ranks.

In [7]:

```
off_rank_points = [[0]*len(seasons) for x in range(len(teams))]
def_rank_points = [[0]*len(seasons) for x in range(len(teams))]

for i in range(len(teams)):
    for j in range(len(seasons)):
        off_rank_points[i][j] = team_score_off_rank['total'].loc[seasons[j], teams[i]]
        def_rank_points[i][j] = team_score_def_rank['total'].loc[seasons[j], teams[i]]

off_rank_points, def_rank_points = pd.DataFrame(off_rank_points, index = teams), pd.DataFrame(def_rank_points, index = teams)
off_rank_points.rename(columns=lambda x: x + 2002, inplace=True)
def_rank_points.rename(columns=lambda x: x + 2002, inplace=True)
off_rank = off_rank_points.rank(axis = 0, method = 'max', ascending = False)
def_rank = def_rank_points.rank(axis = 0, method = 'max', ascending = True)

offrank_pps = pd.DataFrame(index = seasons)
for i in range(len(teams)):
    offrank_pps[i + 1] = pd.DataFrame(amax(off_rank_points[off_rank == amin(off_rank + i, axis = 0)], axis = 0))

defrank_pps = pd.DataFrame(index = seasons)
for i in range(len(teams)):
    defrank_pps[i + 1] = pd.DataFrame(amax(def_rank_points[def_rank == amin(def_rank + i, axis = 0)], axis = 0))
```

Here we bring in the team rankings into the main data frame.

In [8]:

```
off_rank = pd.DataFrame(off_rank.stack())
def_rank = pd.DataFrame(def_rank.stack())
off_rank.columns = ['offrank']
```

```
def_rank.columns = ['defrank']
pbp_data = pbp_data.join(off_rank, on = ['off', 'season'], how = 'left')
pbp_data = pbp_data.join(def_rank, on = ['def', 'season'], how = 'left')
```

Here we create dummy variables and factor variables concerning the ranking of teams. This will help us run a logistic regression using team ranking as covariates.

In [9]:

```
pbp_data['offrankbucket'] = 'NA'
pbp_data['defrankbucket'] = 'NA'
pbp_data['offrank1t4'] = 0
pbp_data['offrank5t30'] = 0
pbp_data['offrank31t32'] = 0
pbp_data['defrank1t4'] = 0
pbp_data['defrank5t30'] = 0
pbp_data['defrank31t32'] = 0
pbp_data.offrankbucket[pbp_data.offrank >= 1], pbp_data.defrankbucket[pbp_data.defrank >= 1] = '(1-4)', '(1-4)'
pbp_data.offrankbucket[pbp_data.offrank >= 5], pbp_data.defrankbucket[pbp_data.defrank >= 5] = '(5-30)', '(5-30)'
pbp_data.offrankbucket[pbp_data.offrank >= 31], pbp_data.defrankbucket[pbp_data.defrank >= 31] = '(31-32)', '(31-32)'
pbp_data.offrank1t4[pbp_data.offrankbucket == '(1-4)'], pbp_data.defrank1t4[pbp_data.defrankbucket == '(1-4)'] = 1, 1
pbp_data.offrank5t30[pbp_data.offrankbucket == '(5-30)'], pbp_data.defrank5t30[pbp_data.defrankbucket == '(5-30)'] = 1, 1
pbp_data.offrank31t32[pbp_data.offrankbucket == '(31-32)'], pbp_data.defrank31t32[pbp_data.defrankbucket == '(31-32)'] = 1, 1
pbp_data['offrankMid'] = pbp_data['offrank5t30']*pbp_data['offrank']
pbp_data['defrankMid'] = pbp_data['defrank5t30']*pbp_data['defrank']
offrankbucket = pd.Categorical.from_array(pbp_data['offrankbucket'])
defrankbucket = pd.Categorical.from_array(pbp_data['defrankbucket'])
pbp_data['offrankbucket'] = offrankbucket.labels
pbp_data['defrankbucket'] = defrankbucket.labels
```

In [10]:

```
pbp_data_new = pbp_data[pbp_data.description.str.contains('kicks') == False]
pbp_data_new.index = arange(len(pbp_data_new))
shifted_data1 = (pbp_data_new.shift(1)).shift(-1)
shifted_data2 = pbp_data_new.shift(1)
shifted_data2.rename(columns=lambda x: 'last' + x, inplace=True)
shifted_data = shifted_data1.join(shifted_data2, how = 'outer')
```

In [11]:

```
offTF = shifted_data['off'] == shifted_data['lastoff']
defTF = shifted_data['def'] == shifted_data['lastdef']
qtrTF = shifted_data['qtr'].isin([3]) & shifted_data['lastqtr'].isin([2]) == True
```

```
shifted_data['condition'] = 'NA'
shifted_data.condition[qtrTF == True] = 1
shifted_data.condition[offTF == False] = 2
shifted_data.condition[defTF == False] = 3
shifted_data.condition[0:30]
down1 = shifted_data[shifted_data['condition'].isin([1, 2, 3])]
```

The following gives drive by drive information. This information is useful in helping us know the expected number of points the offensive team will score given they started the first play of the drive on a specific yard line.

In [12]:

```
firstPlay = down1.index.values
firstPlay = np.array(firstPlay)
lastPlay = firstPlay - 1
lastPlay1 = lastPlay[1:len(lastPlay)]
lastPlay2 = np.append(lastPlay1, (len(pbp_data)-1))
lastPlays = pbp_data_new.ix[lastPlay2]

lastPlays.rename(columns=lambda x: 'last_' + x, inplace=True)
down1['mergeVals'] = np.arange(len(firstPlay))
lastPlays['mergeVals'] = np.arange(len(firstPlay))
driveByDrive = down1.merge(lastPlays, on = 'mergeVals')
driveByDrive['offrankMid'] = driveByDrive['offrank5t30']*driveByDrive['offrank']
driveByDrive['defrankMid'] = driveByDrive['defrank5t30']*driveByDrive['defrank']
```

The following code quantifies the consequences of certain events occurring. We create a vector of the number of points scored at the end of a teams drive. We assume that touchdown plays automatically get seven points, which means we assume the team gets the extra point given they score a touchdown. Then we run a multinomial logistic regression to determine the likelihood of these events based on certain factors, such as team rank and field position. With both pieces of information we determine the expected number points of a team given they convert a first down, their ranking and their field position.

In [13]:

```
driveByDrive['pointsScored'] = 0
madeFG = driveByDrive.last_description.str.contains('GOOD') == True
driveByDrive.pointsScored[madeFG == True] = 3
madeTD = driveByDrive.last_description.str.contains('extra point' or 'Extra Point' or 'Extra point') == True
madeTD2 = driveByDrive.last_description.str.contains('TWO-POINT CONVERSION') == True
driveByDrive.pointsScored[madeTD == True] = 7
driveByDrive.pointsScored[madeTD2 == True] = 7
safety = driveByDrive.last_description.str.contains('SAFETY') == True
driveByDrive.pointsScored[safety == True] = -2
defTD = driveByDrive.last_description.str.contains('TOUCHDOWN') == True
```

```
driveByDrive.pointsScored[defTD == True] = 7
driveByDrive.pointsScored.mean()
driveByDrive['puntReturnForTD'] = 0
puntReturnTD = (driveByDrive.last_description.str.contains('punt') & driveByDrive.last_description.str.contains('TOUCHDOWN')) == True
driveByDrive.puntReturnForTD[puntReturnTD == True] = 1
driveByDrive.pointsScored[puntReturnTD == True] = -7

driveByDrive['interceptionForTD'] = 0
interceptAndTD = driveByDrive.last_description.str.contains('INTERCEPTION') & driveByDrive.last_description.str.contains('TOUCHDOWN') == True
driveByDrive.interceptionForTD[interceptAndTD == True] = 1
driveByDrive.pointsScored[puntReturnTD == True] = -7

driveByDrive['fumbleForTD'] = 0
fumbleAndTD = driveByDrive.last_description.str.contains('FUMBLE') & driveByDrive.last_description.str.contains('TOUCHDOWN') == True
driveByDrive.fumbleForTD[fumbleAndTD == True] = 1
driveByDrive.pointsScored[fumbleAndTD == True] = -7

driveByDrive['blockedPuntForTD'] = 0
blockedPuntAndTD = driveByDrive.last_description.str.contains('BLOCK') & driveByDrive.last_description.str.contains('TOUCHDOWN') == True
driveByDrive.blockedPuntForTD[blockedPuntAndTD == True] = 1
driveByDrive.pointsScored[blockedPuntAndTD == True] = -7

driveByDrive['intercept'] = [1]*len(driveByDrive)
driveByDrive['score'] = [0]*len(driveByDrive)
driveByDrive = pd.DataFrame(driveByDrive)

driveByDrive.score[driveByDrive['pointsScored'] == -7] = 'DefTD'
driveByDrive.score[driveByDrive['pointsScored'] == -2] = 'DefSafety'
driveByDrive.score[driveByDrive['pointsScored'] == 0] = 'NoPoints'
driveByDrive.score[driveByDrive['pointsScored'] == 3] = 'FG'
driveByDrive.score[driveByDrive['pointsScored'] == 7] = 'TD'

driveByDriveNew = (driveByDrive.shift(1)).shift(-1)
driveByDriveShift = driveByDrive[['puntReturnForTD','interceptionForTD','fumbleForTD','blockedPuntForTD']].shift(1)
driveByDriveShift.rename(columns=lambda x: 'remove' + x, inplace=True)
driveByDrive2 = driveByDriveNew.join(driveByDriveShift, how = 'outer')
driveByDrive3 = driveByDrive2[driveByDrive2['removepuntReturnForTD'] != 1]
driveByDrive4 = driveByDrive3[driveByDrive3['removeinterceptionForTD'] != 1]
driveByDrive5 = driveByDrive4[driveByDrive4['removefumbleForTD'] != 1]
driveByDrive6 = driveByDrive5[driveByDrive5['removeblockedPuntForTD'] != 1]
driveByDriveN = driveByDrive6[['score', 'intercept', 'ydline', 'offrank31t32', 'offrankMid', 'defrank31t32', 'defrankMid']]
driveByDriveN = driveByDriveN.dropna(axis = 0, how = "any")
score_logit = sm.MNLogit(driveByDriveN[['score']], driveByDriveN[['intercept'
, 'ydline', 'offrankMid', 'offrank31t32', 'defrankMid', 'defrank31t32']])

score_logitResult = score_logit.fit()
score_logitResult.summary()
```

```
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:506: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future

  start_params = np.zeros((self.K * (self.J-1)))
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
```

```
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w

ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py

:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:514: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  mnfit.params = mnfit.params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:506: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  start_params = np.zeros((self.K * (self.J-1)))
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')

//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
```

```
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future

  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:514: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  mnfit.params = mnfit.params.reshape(self.K, -1, order='F')

Optimization terminated successfully.
         Current function value: 0.872453
         Iterations 12

//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
```

Out[13]:

MNLogit Regression Results

Dep. Variable:	score	**No. Observations:**	65105
Model:	MNLogit	**Df Residuals:**	65081
Method:	MLE	**Df Model:**	20

Date:	Thu, 17 Apr 2014	**Pseudo R-squ.:**	0.05169
Time:	20:05:48	**Log-Likelihood:**	-56801.
converged:	True	**LL-Null:**	-59897.
		LLR p-value:	0.000

score=DefTD	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	17.1021	1.159	14.759	0.000	14.831 19.373
ydline	-0.1892	0.012	-15.208	0.000	-0.214 -0.165
offrankMid	0.0065	0.010	0.620	0.535	-0.014 0.027
offrank31t32	0.0238	0.347	0.069	0.945	-0.656 0.704
defrankMid	0.0103	0.010	1.018	0.309	-0.010 0.030
defrank31t32	0.4581	0.395	1.161	0.246	-0.315 1.231
score=FG	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	23.3837	1.125	20.785	0.000	21.179 25.589
ydline	-0.2381	0.012	-19.834	0.000	-0.262 -0.215
offrankMid	-0.0052	0.009	-0.565	0.572	-0.023 0.013
offrank31t32	-0.7413	0.308	-2.410	0.016	-1.344 -0.139
defrankMid	0.0217	0.009	2.428	0.015	0.004 0.039
defrank31t32	0.3214	0.357	0.901	0.368	-0.378 1.021
score=NoPoints	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	22.7030	1.124	20.200	0.000	20.500 24.906
ydline	-0.2050	0.012	-17.097	0.000	-0.228 -0.181
offrankMid	0.0080	0.009	0.870	0.384	-0.010 0.026
offrank31t32	-0.2494	0.303	-0.823	0.411	-0.844 0.345
defrankMid	0.0104	0.009	1.174	0.241	-0.007 0.028
defrank31t32	0.0488	0.354	0.138	0.890	-0.644 0.742
score=TD	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	23.6977	1.125	21.070	0.000	21.493 25.902
ydline	-0.2343	0.012	-19.524	0.000	-0.258 -0.211
offrankMid	-0.0209	0.009	-2.270	0.023	-0.039 -0.003
offrank31t32	-1.3770	0.307	-4.487	0.000	-1.979 -0.775
defrankMid	0.0302	0.009	3.393	0.001	0.013 0.048
defrank31t32	0.7186	0.355	2.023	0.043	0.023 1.415

```
In [14]: score = score_logitResult.params
         score.columns = ["DefTD","FG","NoPoints","TD"]
         score
```

Out[14]:

	DefTD	FG	NoPoints	TD
intercept	17.102051	23.383689	22.702956	23.697687
ydline	-0.189247	-0.238087	-0.204967	-0.234290
offrankMid	0.006484	-0.005221	0.007977	-0.020917
offrank31t32	0.023844	-0.741291	-0.249409	-1.376970
defrankMid	0.010301	0.021678	0.010377	0.030189
defrank31t32	0.458122	0.321432	0.048783	0.718639

6 rows × 4 columns

The following is code concerning the decision to punt. Here, we find all punt plays and using logistic regression we determine the probability of events happening given that the team making the decision punted the ball at a certain yard line. The other team receiveing the punt can then score an offensive touchdown or field goal, get no points or give up a defensive touchdown or safety.

```
In [15]: driveByDriveNew2 = (driveByDrive.shift(1)).shift(-1)
         driveByDriveShift2 = driveByDrive[['qtr', 'ydline', 'last_description','last_
         ydline']].shift(1)
         driveByDriveShift2.rename(columns=lambda x: 'remove' + x, inplace=True)
         driveByDriveNew3 = driveByDriveNew2.join(driveByDriveShift2, how = 'outer')
         diffHalf = driveByDriveNew3.removeqtr.isin([2]) & driveByDriveNew3.qtr.isin([
         3]) == True
         diffGame = driveByDriveNew3.removeqtr.isin([4]) & driveByDriveNew3.qtr.isin([
         1]) == True
         driveByDriveNew4 = driveByDriveNew3[(diffHalf & diffGame) == False]
         prevPunt = driveByDriveNew4.removelast_description.str.contains('punts')
         driveByDrivePunt = driveByDriveNew4[prevPunt == True]
         driveByDrivePunt2 = driveByDrivePunt[['score', 'intercept', 'removelast_ydlin
         e', 'offrankMid','offrank31t32', 'defrankMid', 'defrank31t32']]
         driveByDrivePunt3 = driveByDrivePunt2.dropna(axis = 0, how = "any")
         score_logit_punt = sm.MNLogit(driveByDrivePunt3[['score']], driveByDrivePunt3
         [['intercept', 'removelast_ydline', 'offrankMid','offrank31t32', 'defrankMid'
         , 'defrank31t32']])

         score_logit_puntResult = score_logit_punt.fit()
         score_logit_puntResult.summary()
```

```
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
```

```
:506: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  start_params = np.zeros((self.K * (self.J-1)))
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
```

```
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py

:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:514: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  mnfit.params = mnfit.params.reshape(self.K, -1, order='F')
```

```
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')

//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:506: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  start_params = np.zeros((self.K * (self.J-1)))
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
```

```
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future

  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1562: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1628: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future
  params = params.reshape(self.K, -1, order='F')
//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:514: DeprecationWarning: using a non-integer number instead of an integer wi
ll result in an error in the future
  mnfit.params = mnfit.params.reshape(self.K, -1, order='F')

Optimization terminated successfully.
```

```
         Current function value: 0.881161
         Iterations 9

//anaconda/lib/python2.7/site-packages/statsmodels/discrete/discrete_model.py
:1503: DeprecationWarning: using a non-integer number instead of an integer w
ill result in an error in the future

  params = params.reshape(self.K, -1, order='F')
```

Out[15]:

MNLogit Regression Results

Dep. Variable:	score	**No. Observations:**	26271
Model:	MNLogit	**Df Residuals:**	26247
Method:	MLE	**Df Model:**	20
Date:	Thu, 17 Apr 2014	**Pseudo R-squ.:**	0.03018
Time:	20:05:50	**Log-Likelihood:**	-23149.
converged:	True	**LL-Null:**	-23869.
		LLR p-value:	2.247e-293

score=DefTD	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	-2.5052	0.567	-4.415	0.000	-3.617 -1.393
removelast_ydline	0.0502	0.009	5.741	0.000	0.033 0.067
offrankMid	0.0073	0.014	0.520	0.603	-0.020 0.035
offrank31t32	-0.1668	0.431	-0.387	0.699	-1.011 0.678
defrankMid	0.0189	0.013	1.431	0.153	-0.007 0.045
defrank31t32	0.2559	0.592	0.432	0.666	-0.905 1.417
score=FG	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	-2.0875	0.461	-4.532	0.000	-2.990 -1.185
removelast_ydline	0.0874	0.007	11.719	0.000	0.073 0.102
offrankMid	-0.0018	0.012	-0.154	0.878	-0.024 0.021
offrank31t32	-0.9428	0.348	-2.712	0.007	-1.624 -0.262
defrankMid	0.0284	0.011	2.621	0.009	0.007 0.050
defrank31t32	0.6024	0.488	1.235	0.217	-0.354 1.558
score=NoPoints	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	1.3069	0.451	2.897	0.004	0.423 2.191
removelast_ydline	0.0607	0.007	8.239	0.000	0.046 0.075
offrankMid	0.0132	0.011	1.160	0.246	-0.009 0.036

offrank31t32	-0.3978	0.338	-1.176	0.240	-1.061 0.265
defrankMid	0.0149	0.011	1.395	0.163	-0.006 0.036
defrank31t32	0.2583	0.482	0.536	0.592	-0.687 1.204
score=TD	**coef**	**std err**	**z**	**P>|z|**	**[95.0% Conf. Int.]**
intercept	-1.1274	0.457	-2.469	0.014	-2.022 -0.233
removelast_ydline	0.0802	0.007	10.814	0.000	0.066 0.095
offrankMid	-0.0164	0.011	-1.428	0.153	-0.039 0.006
offrank31t32	-1.4860	0.346	-4.295	0.000	-2.164 -0.808
defrankMid	0.0381	0.011	3.529	0.000	0.017 0.059
defrank31t32	1.1201	0.485	2.311	0.021	0.170 2.070

In [16]:

```
Punt = pd.DataFrame(score_logit_puntResult.params)
Punt.columns = ["DefTD","FG","NoPoints","TD"]
Punt
```

Out[16]:

	DefTD	**FG**	**NoPoints**	**TD**
intercept	-2.505152	-2.087459	1.306882	-1.127434
removelast_ydline	0.050217	0.087383	0.060661	0.080226
offrankMid	0.007299	-0.001777	0.013215	-0.016401
offrank31t32	-0.166807	-0.942778	-0.397772	-1.485986
defrankMid	0.018914	0.028430	0.014893	0.038073
defrank31t32	0.255890	0.602383	0.258313	1.120144

6 rows × 4 columns

The following code is used to pull out fourth down plays from the data. This is important since we'll use plays from these downs to find the expected number of points given field goal attempt, punt attempt, or go for it attempt.

In [17]:

```
fourthDown = pbp_data[pbp_data['down'] == 4]
fourthPlayNumbers = fourthDown.index
fourthPlayNumbers2 = fourthPlayNumbers[1:len(fourthPlayNumbers)]
fourthPlayNumbers2 = np.append(fourthPlayNumbers2, 0)
nonPenaltyFourthDown = fourthPlayNumbers[fourthPlayNumbers2 - fourthPlayNumbe
rs != 1]
fourthDownPlays = pbp_data.ix[nonPenaltyFourthDown, :]
```

Pulling out field goal attempt data and running a logistic regression to determine the likelihood of converting depending on the yardline the field goal is attempted from.

In [18]:
```
fourth = fourthDownPlays
cond = fourthDownPlays['description'].str.contains('field goal' or 'Field Goa
l' or 'field Goal' or 'Field goal' or 'FIELD GOAL')
field_goal_data = fourthDownPlays[cond == True]
field_goal_data['converted'] = field_goal_data['description'].str.contains('G
OOD')
field_goal_data['distance'] = field_goal_data['ydline'] + 18
field_goal = field_goal_data[['ydline', 'distance', 'description', 'converted
']]
field_goal['intercept'] = [1]*len(field_goal)
field_goal_logit = sm.Logit(field_goal[['converted']], field_goal[['intercept
','ydline']])
field_goal_result = field_goal_logit.fit()
field_goal_result.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.410096
         Iterations 7
```

Out[18]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	9650
Model:	Logit	**Df Residuals:**	9648
Method:	MLE	**Df Model:**	1
Date:	Thu, 17 Apr 2014	**Pseudo R-squ.:**	0.1206
Time:	20:05:51	**Log-Likelihood:**	-3957.4
converged:	True	**LL-Null:**	-4500.1
		LLR p-value:	5.246e-238

	coef	std err	z	P>|z|	[95.0% Conf. Int.]
intercept	3.6030	0.083	43.532	0.000	3.441 3.765
ydline	-0.0981	0.003	-29.699	0.000	-0.105 -0.092

In [19]:
```
fg_vec = [3.603047, -0.098109]
```

The following pulls out the rankings of teams and yards to go to convert on the fourth down. We also perform a logistic regression to determine the likelihood of converting given the rankings and yards to go.

In [20]:

```
cond2 = fourthDownPlays['description'].str.contains('punt' or 'PUNT' or 'Punt')
punt_data = fourthDownPlays[cond2 == True]
punt_data['ydlineotherteam'] = 100 - (punt_data['ydline']-39)
punt_data_final = punt_data[['description', 'ydline', 'ydlineotherteam']]
go = fourthDownPlays[cond == False]
cond2 = go['description'].str.contains('punt' or 'PUNT' or 'Punt')
go_for_it = go[cond2 == False]
play_after_going_for_it = pbp_data.ix[np.array(go_for_it.index) + 1, :]
play_after_gfi_important_vars = play_after_going_for_it[['ydline','offrank', 'defrank', 'offrankMid', 'offrank31t32']]
play_after_gfi_important_vars.rename(columns=lambda x: 'next_' + x, inplace=True)
play_after_gfi_important_vars.index = go_for_it.index
go_for_it_2 = go_for_it.join(play_after_gfi_important_vars)
go_for_it_2['converted'] = go_for_it_2['offrank'] == go_for_it_2['next_offrank']
go_for_it_final = go_for_it_2[['offrank', 'defrank', 'togo', 'offrankMid', 'offrank31t32', 'defrankMid', 'defrank31t32', 'converted']]
go_for_it_final['intercept'] = [1]*len(go_for_it_final)
go_for_it_final = go_for_it_final.dropna(axis = 0, how = "any")
go_for_it_logit = sm.Logit(go_for_it_final[['converted']], go_for_it_final[['intercept','togo', 'offrankMid', 'offrank31t32', 'defrankMid', 'defrank31t32']])
go_for_it_result = go_for_it_logit.fit()
go_for_it_result.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.653088
         Iterations 5

//anaconda/lib/python2.7/site-packages/pandas/core/frame.py:2175: SettingWithCopyWarning: A value is trying to be set on a copy of a slice from a DataFrame
  **kwargs)
```

Out[20]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	5494
Model:	Logit	**Df Residuals:**	5488
Method:	MLE	**Df Model:**	5
Date:	Thu, 17 Apr 2014	**Pseudo R-squ.:**	0.05691
Time:	20:05:51	**Log-Likelihood:**	-3588.1
converged:	True	**LL-Null:**	-3804.6
		LLR p-value:	2.247e-91

	coef	std err	z	P>|z|	[95.0% Conf. Int.]
intercept	0.5808	0.082	7.067	0.000	0.420 0.742
togo	-0.1199	0.007	-17.664	0.000	-0.133 -0.107
offrankMid	-0.0076	0.003	-2.358	0.018	-0.014 -0.001
offrank31t32	-0.4553	0.122	-3.730	0.000	-0.695 -0.216
defrankMid	0.0112	0.003	3.510	0.000	0.005 0.017
defrank31t32	0.4841	0.135	3.578	0.000	0.219 0.749

In [21]:

```
gfi_vec = [0.580768, -0.119942, -0.007568, -0.455296, 0.011190, 0.484075]
```

The following functions will be used to determine the choice to be made given yard line, yards to go, and both offensive and defensive rankings of the team making the decision as well as the other team.

In [22]:

```
def init(ydline, ydtogo, oorank, odrank, ddrank, dorank):
    """This function initiates vectors that will later be used to calculate probabilities and expections based on the information provided"""
    oorank1 = 2 if 5 <= oorank <= 30 else 3 if 31 <= oorank <= 32 else 0
    Ioorank = 1 if oorank1 == 0 else 1 if oorank1 == 3 else oorank
    ddrank1 = 4 if 5 <= ddrank <= 30 else 5 if 31 <= ddrank <= 32 else 0
    Iddrank = 1 if ddrank1 == 0 else 1 if ddrank1 == 3 else ddrank
    odrank1 = 4 if 5 <= odrank <= 30 else 5 if 31 <= odrank <= 32 else 0
    Iodrank = 1 if odrank1 == 0 else 1 if odrank1 == 3 else odrank
    dorank1 = 2 if 5 <= dorank <= 30 else 3 if 31 <= dorank <= 32 else 0
    Idorank = 1 if dorank1 == 0 else 1 if dorank1 == 3 else dorank
    X_off = [1, ydline - ydtogo, 0, 0, 0, 0]; X_off[oorank1] = Ioorank; X_off[ddrank1] = Iddrank
    X_def_score = [1, 100 - (ydline - ydtogo)*(2/5), 0, 0, 0, 0]; X_def_score[dorank1] = Idorank; X_def_score[odrank1] = Iodrank
    X_def_gfi_fail = [1, 100 - ydline, 0, 0, 0, 0]; X_def_gfi_fail[dorank1] = Idorank; X_def_gfi_fail[odrank1] = Iodrank
    X_def_20 = [1, 80, 0, 0, 0, 0]; X_def_20[dorank1] = Idorank; X_def_20[odrank1] = Iodrank
    X_def_fg_fail = [1, 93 - ydline, 0, 0, 0, 0]; X_def_fg_fail[dorank1] = Idorank; X_def_fg_fail[odrank1] = Iodrank
    X_punt = [1, ydline, 0, 0, 0, 0]; X_punt[dorank1] = Idorank; X_punt[odrank1] = Iodrank
    X = {'off': X_off, 'def_score': X_def_score, 'gfi_fail': X_def_gfi_fail, '20': X_def_20, 'fg_fail': X_def_fg_fail, 'punt': X_punt}
    return X
```

In [23]:

```
def prob(ydline, ydtogo, oorank, odrank, ddrank, dorank):
    """This function finds the probability of converting a first down and con
verting a field goal under certain conditions"""
    oorank1 = 2 if 5 <= oorank <= 30 else 3 if 31 <= oorank <= 32 else 0
    Ioorank = 1 if oorank1 == 0 else 1 if oorank1 == 3 else oorank
    ddrank1 = 4 if 5 <= ddrank <= 30 else 5 if 31 <= ddrank <= 32 else 0
    Iddrank = 1 if ddrank1 == 0 else 1 if ddrank1 == 3 else ddrank
    oorank_gfi = 2 if 5 <= oorank <= 30 else 3 if 31 <= oorank <= 32 else 0
    ddrank_gfi = 4 if 5 <= ddrank <= 30 else 5 if 31 <= ddrank <= 32 else 0
    vec = [1, ydtogo, 0, 0, 0, 0]; vec[oorank_gfi] = Ioorank; vec[ddrank_gfi]
 = Iddrank
    fg = exp((fg_vec[0] + ydline*(fg_vec[1])))/(1 + exp((fg_vec[0] + ydline*(
fg_vec[1]))))
    gfi = exp((vec[0]*gfi_vec[0] + vec[1]*(gfi_vec[1]) + vec[2]*(gfi_vec[2])
* vec[3]*(gfi_vec[3]) + vec[4]*(gfi_vec[4]) + vec[5]*(gfi_vec[5])))/(1 + exp(
vec[0]*gfi_vec[0] + vec[1]*(gfi_vec[1]) + vec[2]*(gfi_vec[2]) + vec[3]*(gfi_v
ec[3]) + vec[4]*(gfi_vec[4]) + vec[5]*(gfi_vec[5])))
    X = {'fg': fg, 'gfi': gfi}
    return X
```

In [24]:

```
def log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, vec):
    """Using the results of one of the logistic regressions from above, value
s are calculated that will be used later to help in determining the expectati
on of points given go for it, punt or field goal decision."""
    Xsum = (1 + exp(sum(vec*score.ix[:,0])) + exp(sum(vec*score.ix[:,1])) + e
xp(sum(vec*score.ix[:,2])) + exp(sum(vec*score.ix[:,3])))
    DefTD = exp(sum(vec*score.ix[:,0]))/Xsum
    FG = exp(sum(vec*score.ix[:,1]))/Xsum
    NoPoints = exp(sum(vec*score.ix[:,2]))/Xsum
    TD = exp(sum(vec*score.ix[:,3]))/Xsum
    DefSafety = 1/Xsum
    X = {'DefTD': DefTD, 'FG': FG, 'NoPoints': NoPoints, 'TD': TD, 'DefSafety
': DefSafety}
    return X
```

In [25]:

```
def log_punt(ydline, ydtogo, oorank, odrank, ddrank, dorank, vec):
    """Using the results of one of the logistic regressions from above, value
s are calculated that will be used later to help in determining the expectati
on of points given go for it, punt or field goal decision."""
    Xsum = (1 + exp(sum(vec*Punt.ix[:,0])) + exp(sum(vec*Punt.ix[:,1])) + exp
(sum(vec*Punt.ix[:,2])) + exp(sum(vec*Punt.ix[:,3])))
    DefTD = exp(sum(vec*Punt.ix[:,0]))/Xsum
    FG = exp(sum(vec*Punt.ix[:,1]))/Xsum
    NoPoints = exp(sum(vec*Punt.ix[:,2]))/Xsum
    TD = exp(sum(vec*Punt.ix[:,3]))/Xsum
    DefSafety = 1/Xsum
    X = {'DefTD': DefTD, 'FG': FG, 'NoPoints': NoPoints, 'TD': TD, 'DefSafety
': DefSafety}
```

```
    return X
```

In [26]:

```
def gfi_expect(ydline, ydtogo, oorank, odrank, ddrank, dorank):
    """Calculates the point expectation given the offensive team goes for the first down."""
    x = init(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    y = prob(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    X20 = log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['20'])
    XDS = log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['def_score'])
    XGFI = log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['gfi_fail'])
    XOFF = log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['off'])

    if ydline <= ydtogo:
        E_gfi = y['gfi']*(7 - (X20['DefTD']*(-7) + X20['FG']*(3) + X20['TD']*(7) + X20['DefSafety']*(-2))) - (1 - y['gfi'])*(XGFI['DefTD']*(-7) + XGFI['FG']*(3) + XGFI['TD']*(7) + XGFI['DefSafety']*(-2))
    else:
        E_gfi = y['gfi']*((XOFF['FG']*(3) + XOFF['TD']*(7) + XOFF['DefSafety']*(-2) + XOFF['DefTD']*(-7)) - (XOFF['FG'] + XOFF['TD'] + XOFF['DefSafety'])*(X20['DefTD']*(-7) + X20['FG']*(3) + X20['TD']*(7) + X20['DefSafety']*(-2)) - XOFF['NoPoints']*(XDS['FG']*(3) + XDS['TD']*(7) + XDS['DefSafety']*(-2) + XDS['DefTD']*(-7))) - (1 - y['gfi'])*(XGFI['DefTD']*(-7) + XGFI['FG']*(3) + XGFI['TD']*(7) + XGFI['DefSafety']*(-2))
    return E_gfi
```

In [27]:

```
def fg_expect(ydline, ydtogo, oorank, odrank, ddrank, dorank):
    """Calculates the point expectation given the offensive team goes for a field goal."""
    x = init(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    y = prob(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    X20 = log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['20'])
    XFG = log_score(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['fg_fail'])

    E_fg = y['fg']*(3 - (X20['DefTD']*(-7) + X20['FG']*(3) + X20['TD']*(7) + X20['DefSafety']*(-2))) - (1 - y['fg'])*(XFG['DefTD']*(-7) + XFG['FG']*(3) + XFG['TD']*(7) + XFG['DefSafety']*(-2))
    return E_fg
```

In [28]:

```
def punt_expect(ydline, ydtogo, oorank, odrank, ddrank, dorank):
    """Calculates the point expectation given the offensive team punts."""
    x = init(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    XP = log_punt(ydline, ydtogo, oorank, odrank, ddrank, dorank, x['punt'])
    E_punt = - (XP['DefTD']*(-7) + XP['FG']*(3) + XP['TD']*(7) + XP['DefSafet
y']*(-2))
    return E_punt
```

In [29]:

```
def decision(ydline, ydtogo, oorank, odrank, ddrank, dorank):
    """Returns the three expectations calculated above to help the user deter
mine which decision is best."""
    gfi = gfi_expect(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    fg = fg_expect(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    punt = punt_expect(ydline, ydtogo, oorank, odrank, ddrank, dorank)
    dec = {'Go For It': gfi, 'Field Goal': fg, 'Punt': punt}
    return dec
```

convert_prob is a three dimensional matrix that gives the probability of the offensive team converting a first down given how many yards to go for the first down, the offensive rank of the offense and the defensive rank of the defense.

In [30]:

```
convert_prob = np.ndarray(shape = (10, 32, 32)) #i -> yards to go, j -> offen
sive rank, k -> defensive rank

convert_prob[:][:][:] = 0
convert_vec = [1, 0, 0, 0, 0, 0]
for i in range(0, 10):
    for j in range(0, 4):
        for k in range(0, 4):
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)


convert_vec = [1, 0, 0, 0, 0, 0]
for i in range(0, 10):
    for j in range(0, 4):
        for k in range(4, 30):
            convert_vec[4] = k + 1
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)


convert_vec = [1, 0, 0, 0, 0, 1]
for i in range(0, 10):
    for j in range(0, 4):
        for k in range(30, 32):
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)


convert_vec = [1, 0, 0, 0, 0, 0]
for i in range(0, 10):
    for j in range(4, 30):
        for k in range(0, 4):
            convert_vec[2] = j + 1
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)

convert_vec = [1, 0, 0, 0, 0, 0]
for i in range(0, 10):
    for j in range(4, 30):
        for k in range(4, 30):
            convert_vec[2] = j + 1
            convert_vec[4] = k + 1
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)

convert_vec = [1, 0, 0, 0, 0, 1]
for i in range(0, 10):
    for j in range(4, 30):
        for k in range(30, 32):
            convert_vec[2] = j + 1
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)


convert_vec = [1, 0, 0, 1, 0, 0]
for i in range(0, 10):
    for j in range(30, 32):
        for k in range(0, 4):
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)

convert_vec = [1, 0, 0, 1, 0, 0]
for i in range(0, 10):
    for j in range(30, 32):
        for k in range(4, 30):
            convert_vec[4] = k + 1
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)

convert_vec = [1, 0, 0, 1, 0, 1]
for i in range(0, 10):
    for j in range(30, 32):
        for k in range(30, 32):
            convert_vec[1] = i
            convert_prob[i][j][k] = go_for_it_result.predict(convert_vec, Tru
e)
```

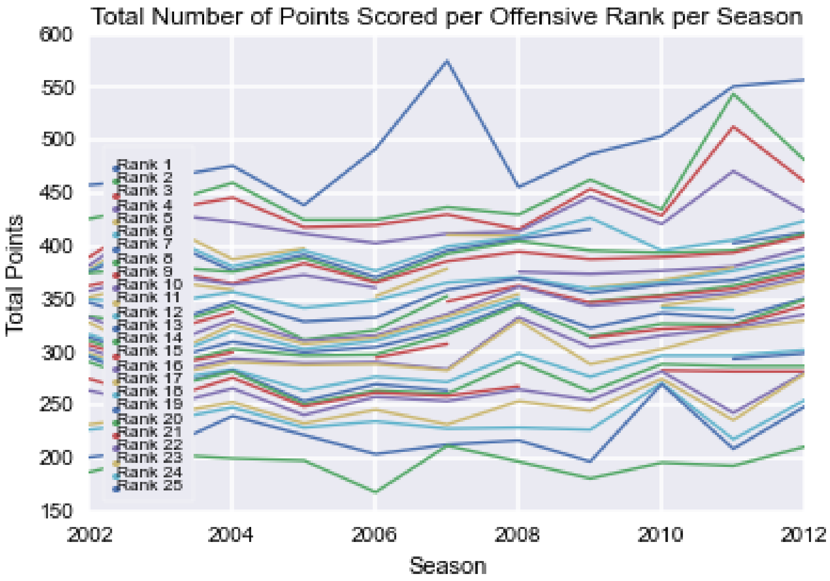
In [31]:

```
z1 = np.array(convert_prob[1])
z2 = np.array(convert_prob[3])
z3 = np.array(convert_prob[6])
z4 = np.array(convert_prob[9])
np.savetxt('z1.txt', z1, delimiter = ',')
np.savetxt('z2.txt', z2, delimiter = ',')
np.savetxt('z3.txt', z3, delimiter = ',')
np.savetxt('z4.txt', z4, delimiter = ',')
```

The following is a plot of total points scored on offense per season per ranked team. Each color represents a different rank.

In [32]:

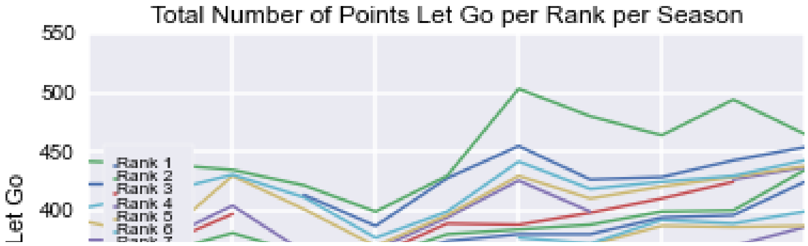
```
fig, ax = plt.subplots()
ax.plot(offrank_pps.index, offrank_pps[1], label = "Rank 1")
ax.plot(offrank_pps.index, offrank_pps[2], label = "Rank 2")
ax.plot(offrank_pps.index, offrank_pps[3], label = "Rank 3")
ax.plot(offrank_pps.index, offrank_pps[4], label = "Rank 4")
ax.plot(offrank_pps.index, offrank_pps[5], label = "Rank 5")
ax.plot(offrank_pps.index, offrank_pps[6], label = "Rank 6")
ax.plot(offrank_pps.index, offrank_pps[7], label = "Rank 7")
ax.plot(offrank_pps.index, offrank_pps[8], label = "Rank 8")
ax.plot(offrank_pps.index, offrank_pps[9], label = "Rank 9")
ax.plot(offrank_pps.index, offrank_pps[10], label = "Rank 10")
ax.plot(offrank_pps.index, offrank_pps[11], label = "Rank 11")
ax.plot(offrank_pps.index, offrank_pps[12], label = "Rank 12")
ax.plot(offrank_pps.index, offrank_pps[13], label = "Rank 13")
ax.plot(offrank_pps.index, offrank_pps[14], label = "Rank 14")
ax.plot(offrank_pps.index, offrank_pps[15], label = "Rank 15")
ax.plot(offrank_pps.index, offrank_pps[16], label = "Rank 16")
ax.plot(offrank_pps.index, offrank_pps[17], label = "Rank 17")
ax.plot(offrank_pps.index, offrank_pps[18], label = "Rank 18")
ax.plot(offrank_pps.index, offrank_pps[19], label = "Rank 19")
ax.plot(offrank_pps.index, offrank_pps[20], label = "Rank 20")
ax.plot(offrank_pps.index, offrank_pps[21], label = "Rank 21")
ax.plot(offrank_pps.index, offrank_pps[22], label = "Rank 22")
ax.plot(offrank_pps.index, offrank_pps[23], label = "Rank 23")
ax.plot(offrank_pps.index, offrank_pps[24], label = "Rank 24")
ax.plot(offrank_pps.index, offrank_pps[25], label = "Rank 25")
ax.plot(offrank_pps.index, offrank_pps[26])
ax.plot(offrank_pps.index, offrank_pps[27])
ax.plot(offrank_pps.index, offrank_pps[28])
ax.plot(offrank_pps.index, offrank_pps[29])
ax.plot(offrank_pps.index, offrank_pps[30])
ax.plot(offrank_pps.index, offrank_pps[31])
ax.plot(offrank_pps.index, offrank_pps[32])
xlabel("Season")
ylabel("Total Points")
title("Total Number of Points Scored per Offensive Rank per Season")
legend = ax.legend(loc=(.02, .02), fontsize = .5, frameon = True, borderpad =
 10)
for label in legend.get_texts():
    label.set_fontsize('small')

for label in legend.get_lines():
    label.set_linewidth(3)

savefig('OffRankperSeason.pdf')
plt.show()
```

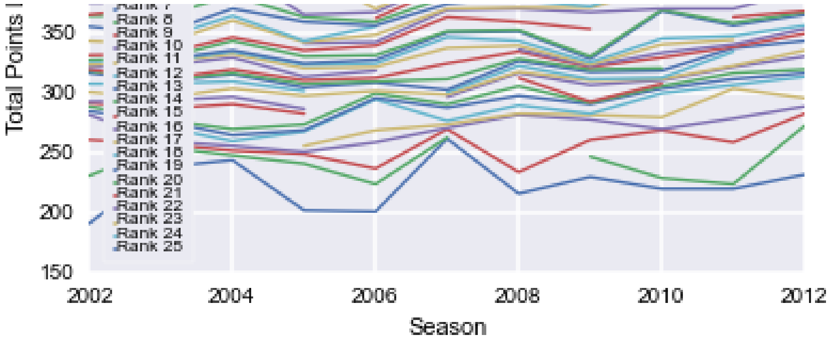


The following is a plot of total points scored on a defense per season per ranked team. Each color represents a different rank.

In [33]:

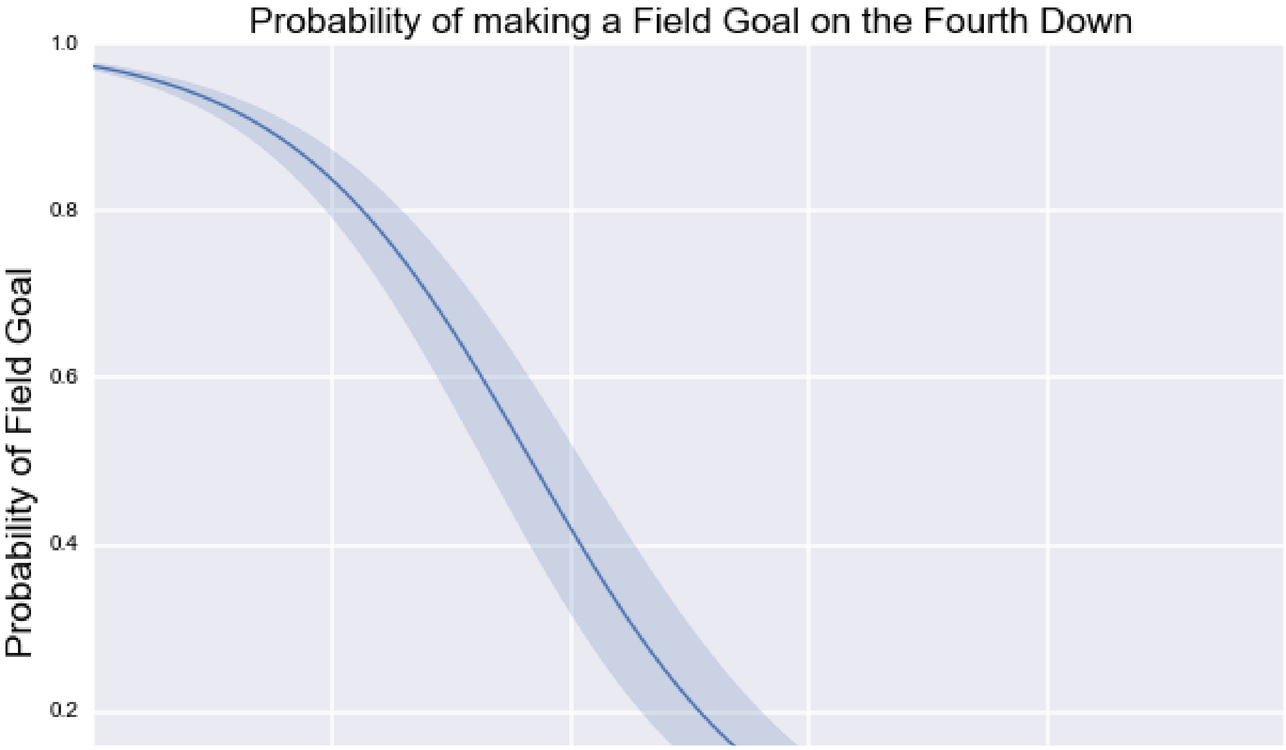
```
fig, ax = plt.subplots()
ax.plot(defrank_pps.index, defrank_pps[1], label = "Rank 1")
ax.plot(defrank_pps.index, defrank_pps[2], label = "Rank 2")
ax.plot(defrank_pps.index, defrank_pps[3], label = "Rank 3")
ax.plot(defrank_pps.index, defrank_pps[4], label = "Rank 4")
ax.plot(defrank_pps.index, defrank_pps[5], label = "Rank 5")
ax.plot(defrank_pps.index, defrank_pps[6], label = "Rank 6")
ax.plot(defrank_pps.index, defrank_pps[7], label = "Rank 7")
ax.plot(defrank_pps.index, defrank_pps[8], label = "Rank 8")
ax.plot(defrank_pps.index, defrank_pps[9], label = "Rank 9")
ax.plot(defrank_pps.index, defrank_pps[10], label = "Rank 10")
ax.plot(defrank_pps.index, defrank_pps[11], label = "Rank 11")
ax.plot(defrank_pps.index, defrank_pps[12], label = "Rank 12")
ax.plot(defrank_pps.index, defrank_pps[13], label = "Rank 13")
ax.plot(defrank_pps.index, defrank_pps[14], label = "Rank 14")
ax.plot(defrank_pps.index, defrank_pps[15], label = "Rank 15")
ax.plot(defrank_pps.index, defrank_pps[16], label = "Rank 16")
ax.plot(defrank_pps.index, defrank_pps[17], label = "Rank 17")
ax.plot(defrank_pps.index, defrank_pps[18], label = "Rank 18")
ax.plot(defrank_pps.index, defrank_pps[19], label = "Rank 19")
ax.plot(defrank_pps.index, defrank_pps[20], label = "Rank 20")
ax.plot(defrank_pps.index, defrank_pps[21], label = "Rank 21")
ax.plot(defrank_pps.index, defrank_pps[22], label = "Rank 22")
ax.plot(defrank_pps.index, defrank_pps[23], label = "Rank 23")
ax.plot(defrank_pps.index, defrank_pps[24], label = "Rank 24")
ax.plot(defrank_pps.index, defrank_pps[25], label = "Rank 25")
ax.plot(defrank_pps.index, defrank_pps[26])
ax.plot(defrank_pps.index, defrank_pps[27])
ax.plot(defrank_pps.index, defrank_pps[28])
ax.plot(offrank_pps.index, defrank_pps[29])
ax.plot(offrank_pps.index, defrank_pps[30])
ax.plot(offrank_pps.index, defrank_pps[31])
ax.plot(offrank_pps.index, defrank_pps[32])
xlabel("Season")
ylabel("Total Points Let Go")
title("Total Number of Points Let Go per Rank per Season")
legend = ax.legend(loc=(.02, .02), fontsize = .5, frameon = True, borderpad = 10)
for label in legend.get_texts():
    label.set_fontsize('small')

for label in legend.get_lines():
    label.set_linewidth(1.5)  # the legend line width

savefig('DefRankperSeason.pdf')
plt.show()
```





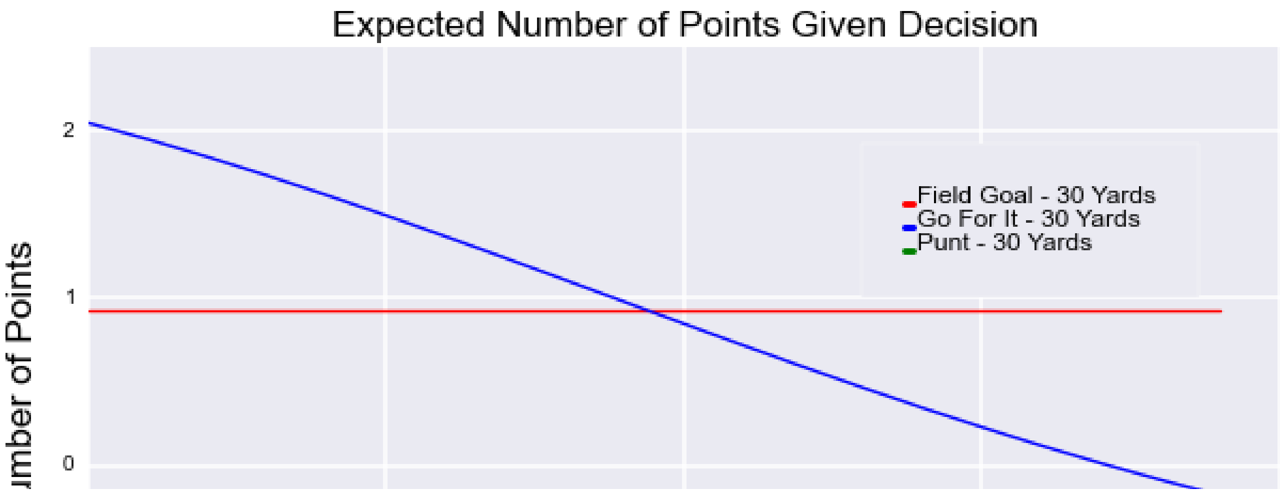
The following graph has probabilities of converting field goals on the y-axis and the yardline the field goal is attempted from. In the midterm presentation the shading on the following graph was purely aesthetic. After creating vectors of the standard errors the following graph now contains the actual confidence interval.

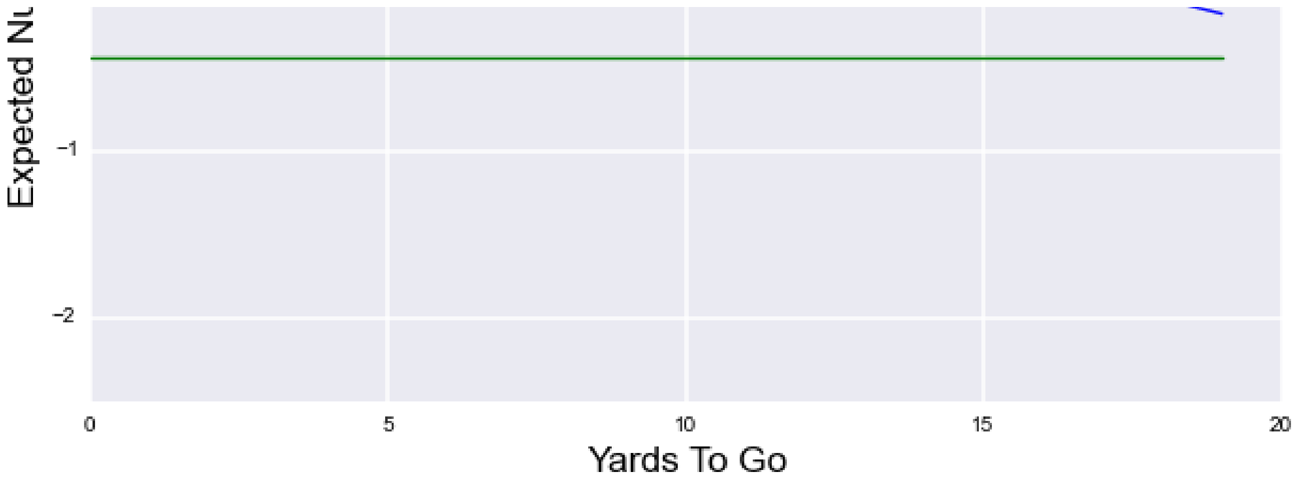
```
In [34]: fg_prob = field_goal_result.predict([[1, x] for x in yard], True)
         upper = [3.765, -0.092]
         lower = [3.441, -0.105]
         prob_upper = [exp(3.765 -0.092*g)/(1 + exp(3.765 -0.092*g)) for g in yard]
         prob_lower = [exp(3.441 -0.105*k)/(1 + exp(3.441 -0.105*k)) for k in yard]
         fig = plt.gcf()
         fig.set_size_inches(10,7)
         plt.plot(yard, np.array(fg_prob), linestyle = '-')
         c1 = sns.color_palette("deep", 2)
         plt.fill_between(yard, prob_lower, prob_upper, color=c1, alpha=.2)
         xlabel('Yard Line', size = "xx-large")
         ylabel('Probability of Field Goal', size = "xx-large")
         title('Probability of making a Field Goal on the Fourth Down', size = "xx-large")
         savefig('fieldgoalprob.pdf')
```



The following is the plot of the decision that should be made from the 30 yard line, where the offensive team has an outstanding ranking and the defensive team is ranked poorly.

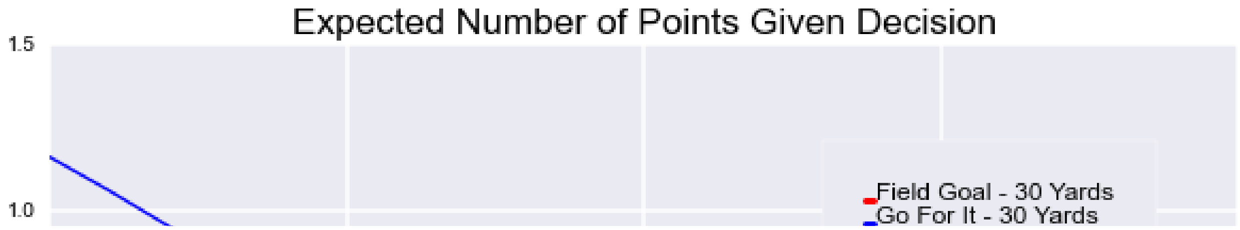
In [35]:
```
x = range(20)
d2 = [decision(30, k, 1, 1, 25, 25) for k in x]
d2 = pd.DataFrame(d2)
fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
ax.plot(d2.index, d2['Field Goal'], color = 'r', label = "Field Goal - 30 Yards", linestyle = '-')
ax.plot(d2.index, d2['Go For It'], color = 'b', label = "Go For It - 30 Yards", linestyle = '-')
ax.plot(d2.index, d2['Punt'], color = 'g', label = "Punt - 30 Yards", linestyle = '-')
xlabel("Yards To Go", size = "xx-large")
ylabel("Expected Number of Points", size = "xx-large")
title("Expected Number of Points Given Decision", size = "xx-large")
legend = ax.legend(loc=(.65, .7), fontsize = 2, frameon = True, borderpad = 10)
ylim(ymax = 2.5, ymin = -2.5)
for label in legend.get_texts():
    label.set_fontsize('large')

for label in legend.get_lines():
    label.set_linewidth(3)

savefig('Decision30112525.pdf')
plt.show()
```

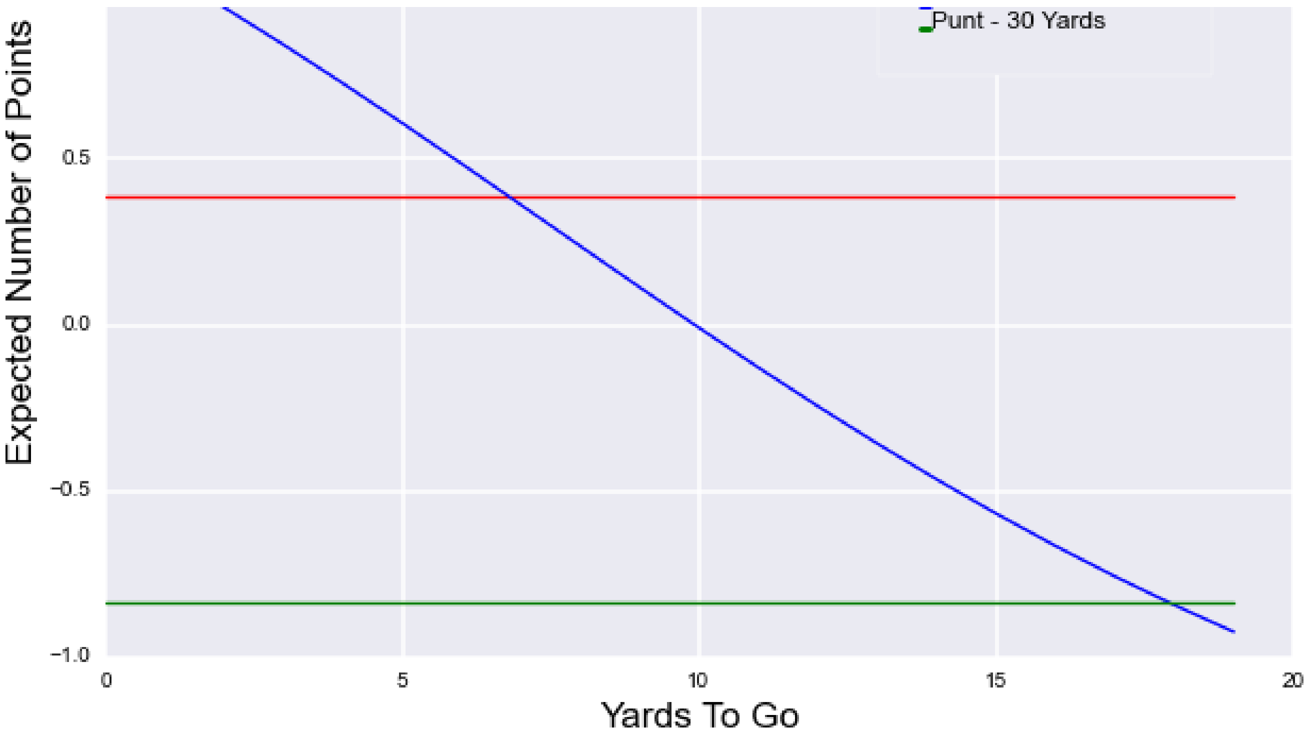




The following is the plot of the decision that should be made from the 30 yard line, where the offensive team has an mediocre ranking and the defensive team is ranked mediocre.

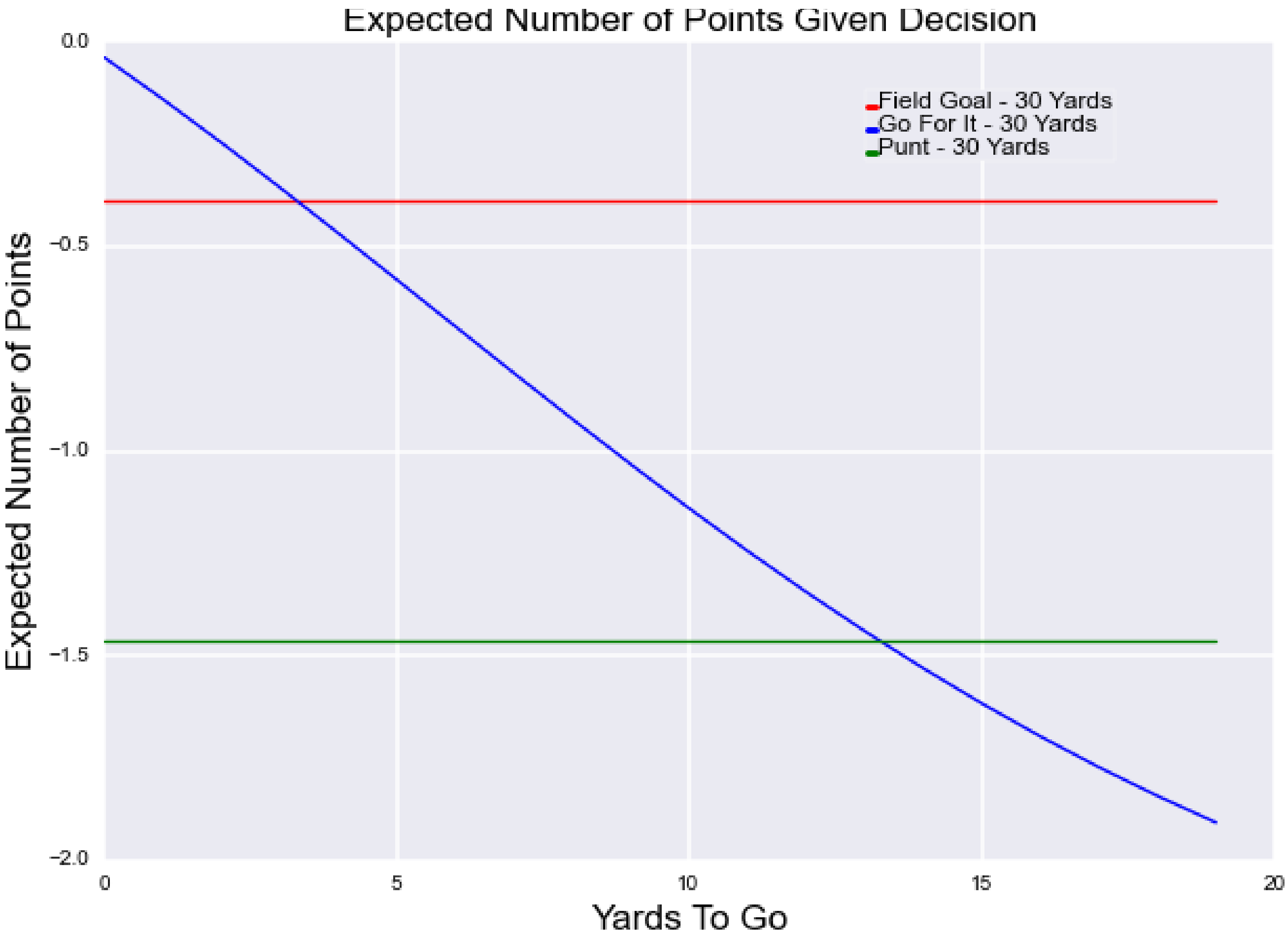
```
In [36]: x = range(20)
d4 = [decision(30, k, 15, 15, 15, 15) for k in x]
d4 = pd.DataFrame(d4)
fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
ax.plot(d4.index, d4['Field Goal'], color = 'r', label = "Field Goal - 30 Yards", linestyle = '-')
ax.plot(d4.index, d4['Go For It'], color = 'b', label = "Go For It - 30 Yards", linestyle = '-')
ax.plot(d4.index, d4['Punt'], color = 'g', label = "Punt - 30 Yards", linestyle = '-')
xlabel("Yards To Go", size = "xx-large")
ylabel("Expected Number of Points", size = "xx-large")
title("Expected Number of Points Given Decision", size = "xx-large")

legend = ax.legend(loc=(.65, .7), fontsize = 2, frameon = True, borderpad = 10)
for label in legend.get_texts():
    label.set_fontsize('large')

for label in legend.get_lines():
    label.set_linewidth(3)  # the legend line width

savefig('Decision3015151515.pdf')
plt.show()
```





The following is the plot of the decision that should be made from the 30 yard line, where the offensive team has a poor ranking and the defensive team is ranked highly.

```
In [37]: x = range(20)
         d6 = [decision(30, k, 25, 25, 1, 1) for k in x]
         d6 = pd.DataFrame(d6)
         fig, ax = plt.subplots()
         fig.set_size_inches(10, 7)
         ax.plot(d6.index, d6['Field Goal'], color = 'r', label = "Field Goal - 30 Yar
         ds", linestyle = '-')
         ax.plot(d6.index, d6['Go For It'], color = 'b', label = "Go For It - 30 Yards
         ", linestyle = '-')
         ax.plot(d6.index, d6['Punt'], color = 'g', label = "Punt - 30 Yards", linesty
         le = '-')
         xlabel("Yards To Go", size = "xx-large")
         ylabel("Expected Number of Points", size = "xx-large")
         title("Expected Number of Points Given Decision", size = "xx-large")

         legend = ax.legend(loc=(.65, .855), fontsize = 2, frameon = True)
         for label in legend.get_texts():
             label.set_fontsize('large')

         for label in legend.get_lines():
             label.set_linewidth(3)  # the legend line width

         savefig('Decision30252511.pdf')
         plt.show()
```



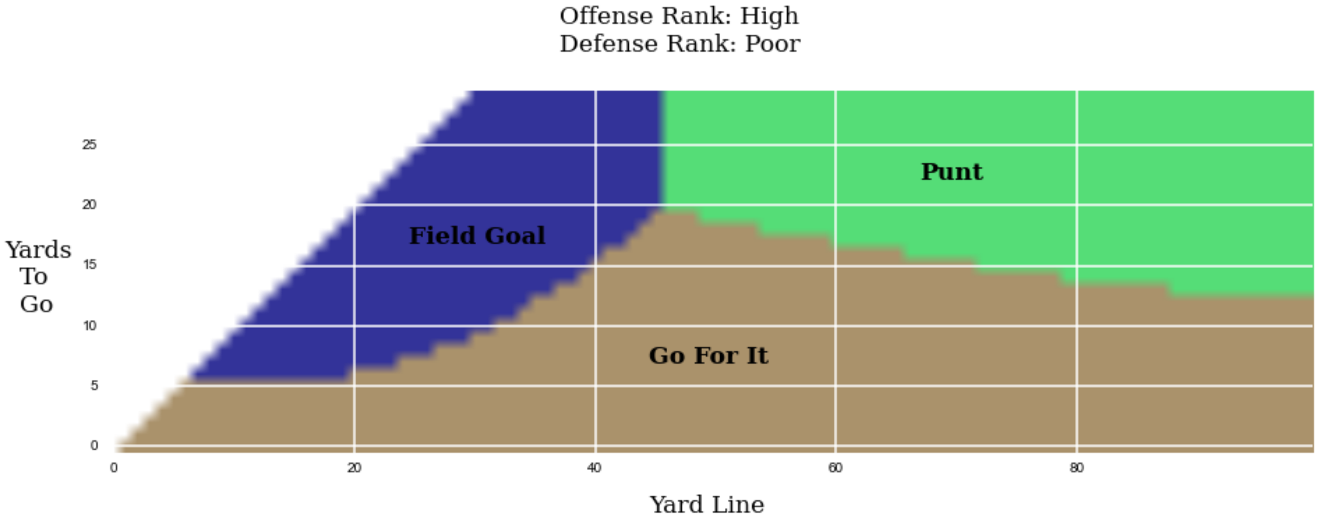
In [38]:

```
def f(x, y, z):
    return z.loc[x, y]
```

The following gives the decision to be made given high ranking of the offensive team and poor ranking of the defensive team.

In [39]:

```
od1 = 1
oo1 = 1
do1 = 25
dd1 = 25
final_decision1 = pd.DataFrame(index = yard)
for j in range(30):
    dec1 = [decision(k, j, oo1, od1, dd1, do1) for k in yard]
    final_decision1[j] = pd.DataFrame([max(g, key = g.get) for g in dec1])

x1 = pd.DataFrame(index = range(100), columns = range(30))
for i in range(30):
    for j in range(100):
        if final_decision1.loc[j,i] == 'Go For It':
            x1.loc[j, i] = 0
        elif final_decision1.loc[j,i] == 'Punt':
            x1.loc[j, i] = 1
        else:
            x1.loc[j, i] = 2

mesh1 = np.meshgrid(np.array(x1.index), np.array(x1.columns))
Z1 = f(np.array(x1.index), np.array(x1.columns), x1)
Z1 = np.array(Z1, dtype = float)
Z1 = Z1.transpose()
for i in range(30):
    for j in range(i + 1):
        Z1[i, j] = -1
plt.axes([1, 1, 2, 2])
plt.imshow(Z1, cmap= 'terrain_r', origin='lower')
figtext(1.9,1.4, "Yard Line", family='serif', size='xx-large')
figtext(0.83,1.9, "Yards\n  To\n  Go", family='serif', size='xx-large')
figtext(1.75,2.55, "Offense Rank: High\nDefense Rank: Poor", family='serif', size='xx-large')
figtext(2.35,2.23, "Punt", family='serif', size='xx-large', color='black', weight = 'bold')
figtext(1.5,2.07, "Field Goal", family='serif', size='xx-large', color = 'black', weight = 'bold')
figtext(1.9,1.77, "Go For It", family='serif', size='xx-large', color = 'black', weight = 'bold')
plt.show()
```

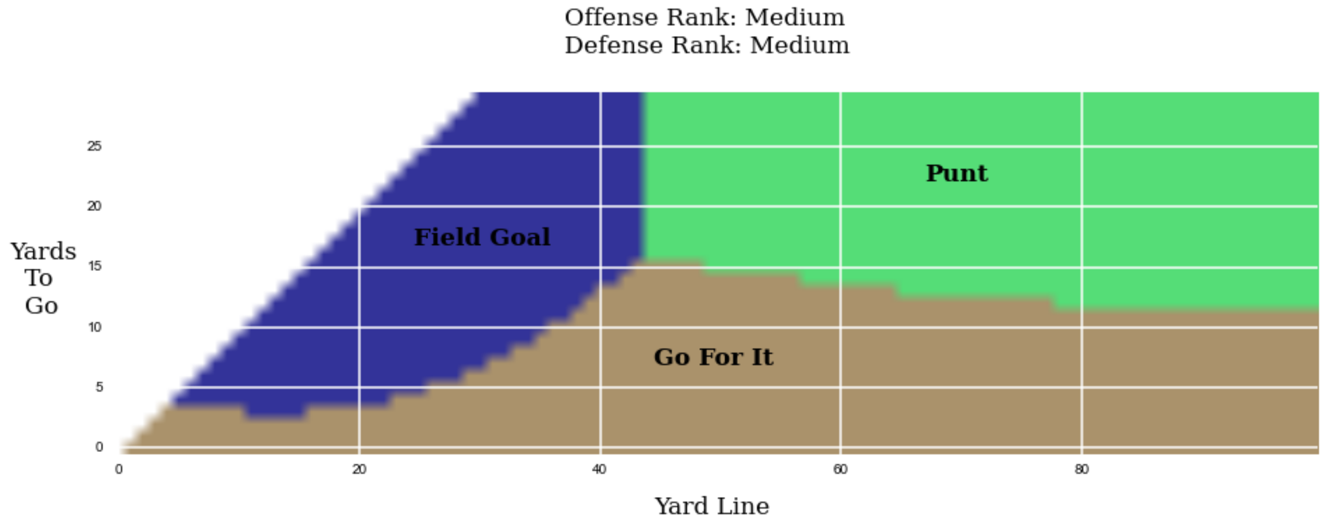


The following gives the decision to be made given mediocre ranking of the offensive team and mediocre ranking of the defensive team.

In [40]:

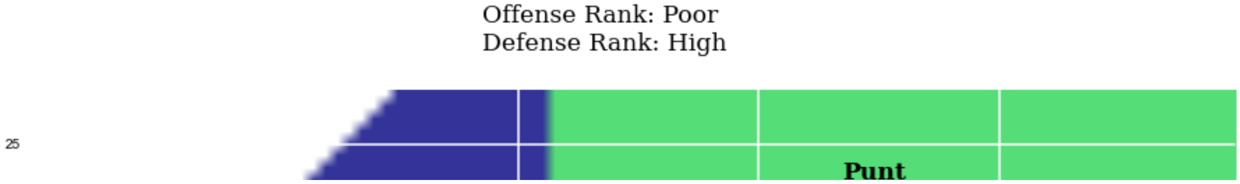
```
od2 = 15
oo2 = 15
do2 = 15
dd2 = 15
final_decision2 = pd.DataFrame(index = yard)
for j in range(30):
```

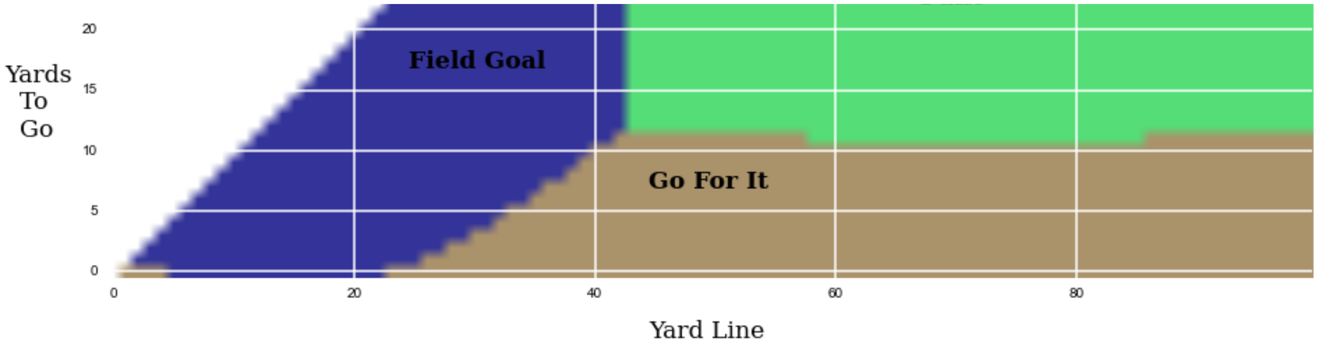
```python
    dec2 = [decision(k, j, oo2, od2, dd2, do2) for k in yard]
    final_decision2[j] = pd.DataFrame([max(g, key = g.get) for g in dec2])

x2 = pd.DataFrame(index = range(100), columns = range(30))
for i in range(30):
    for j in range(100):
        if final_decision2.loc[j,i] == 'Go For It':
            x2.loc[j, i] = 0
        elif final_decision2.loc[j,i] == 'Punt':
            x2.loc[j, i] = 1
        else:
            x2.loc[j, i] = 2

mesh2 = np.meshgrid(np.array(x2.index), np.array(x2.columns))
Z2 = f(np.array(x2.index), np.array(x2.columns), x2)
Z2 = np.array(Z2, dtype = float)
Z2 = Z2.transpose()
for i in range(30):
    for j in range(i + 1):
        Z2[i, j] = -1
plt.axes([1, 1, 2, 2])
plt.imshow(Z2, cmap= 'terrain_r', origin='lower')
figtext(1.9,1.4, "Yard Line", family='serif', size='xx-large')
figtext(0.83,1.9, "Yards\n  To\n  Go", family='serif', size='xx-large')
figtext(1.75,2.55, "Offense Rank: Medium\nDefense Rank: Medium", family='serif', size='xx-large')
figtext(2.35,2.23, "Punt", family='serif', size='xx-large', color='black', weight = 'bold')
figtext(1.5,2.07, "Field Goal", family='serif', size='xx-large', color = 'black', weight = 'bold')
figtext(1.9,1.77, "Go For It", family='serif', size='xx-large', color = 'black', weight = 'bold')
plt.show()
```



The following gives the decision to be made given poor ranking of the offensive team and high ranking of the defensive team.

In [41]:
```
od3 = 25
oo3 = 25
do3 = 2
dd3 = 2
final_decision3 = pd.DataFrame(index = yard)
for j in range(30):
    dec3 = [decision(k, j, oo3, od3, dd3, do3) for k in yard]
    final_decision3[j] = pd.DataFrame([max(g, key = g.get) for g in dec3])

x3 = pd.DataFrame(index = range(100), columns = range(30))
for i in range(30):
    for j in range(100):
        if final_decision3.loc[j,i] == 'Go For It':
            x3.loc[j, i] = 0
        elif final_decision3.loc[j,i] == 'Punt':
            x3.loc[j, i] = 1
        else:
            x3.loc[j, i] = 2

mesh3 = np.meshgrid(np.array(x3.index), np.array(x3.columns))
Z3 = f(np.array(x3.index), np.array(x3.columns), x3)
Z3 = np.array(Z3, dtype = float)
Z3 = Z3.transpose()
for i in range(30):
    for j in range(i + 1):
        Z3[i, j] = -1
plt.axes([1, 1, 2, 2])
plt.imshow(Z3, cmap= 'terrain_r', origin='lower')
figtext(1.9,1.4, "Yard Line", family='serif', size='xx-large')
figtext(0.83,1.9, "Yards\n  To\n  Go", family='serif', size='xx-large')
figtext(1.75,2.55, "Offense Rank: Poor\nDefense Rank: High", family='serif', size='xx-large')
figtext(2.35,2.23, "Punt", family='serif', size='xx-large', color='black', weight = 'bold')
figtext(1.5,2.07, "Field Goal", family='serif', size='xx-large', color = 'black', weight = 'bold')
figtext(1.9,1.77, "Go For It", family='serif', size='xx-large', color = 'black', weight = 'bold')
plt.show()
```

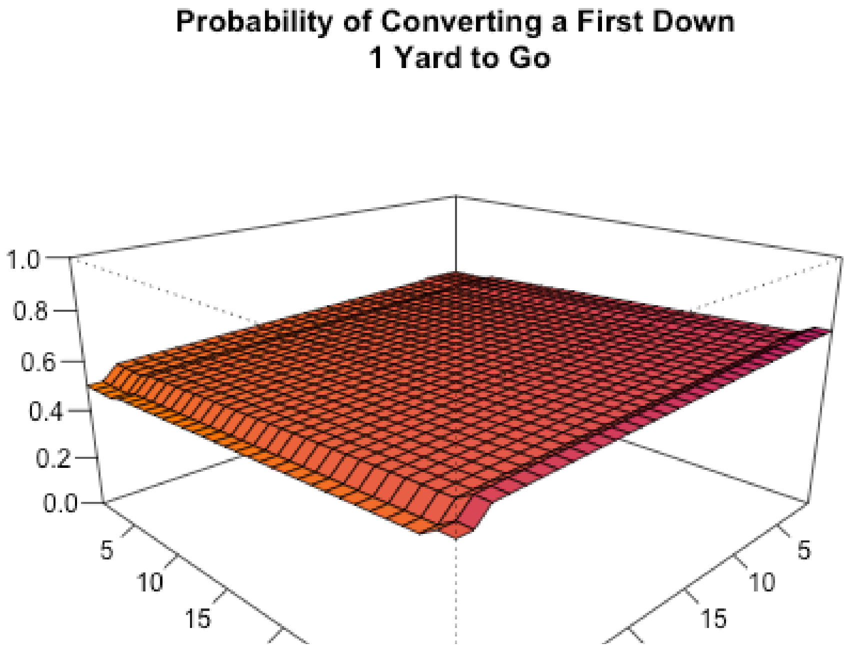


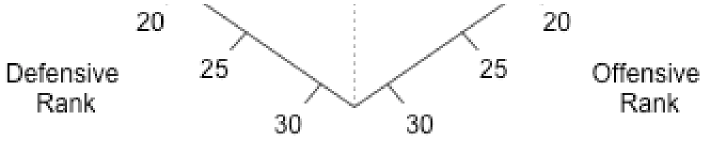


The following shows the probability of conversion of firt down per offensive and defensive rankings with 1 yard to go.

In [42]:

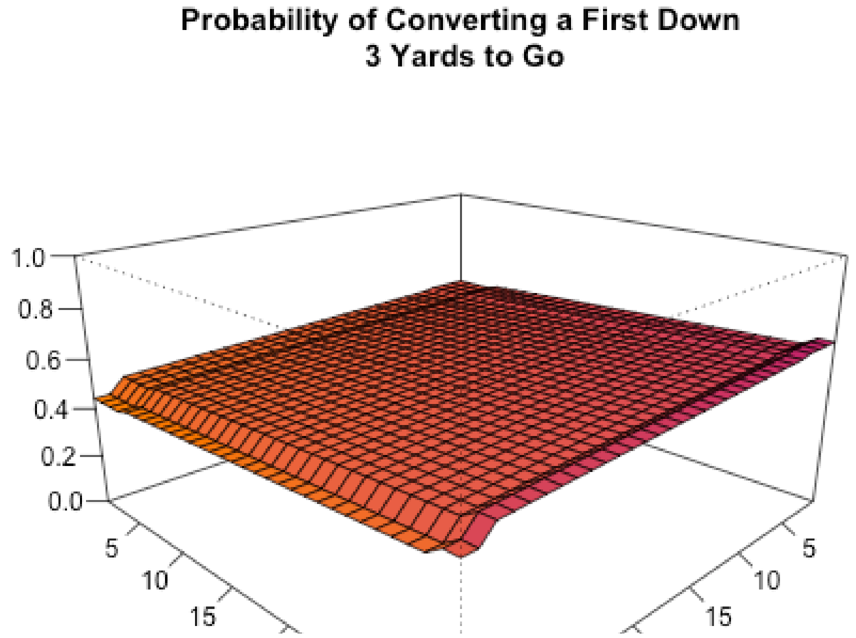
```
%%R
z1 <- read.csv("z1.txt", header = F)
z1 <- as.matrix(z1)
x <- c(1:32); y <- c(1:32)
png("z1plot.png")
persp(x, y, z1, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) + 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 20, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First Down\n 1 Yard to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
dev.off()
persp(x, y, z1, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) + 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 20, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First Down\n 1 Yard to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
```

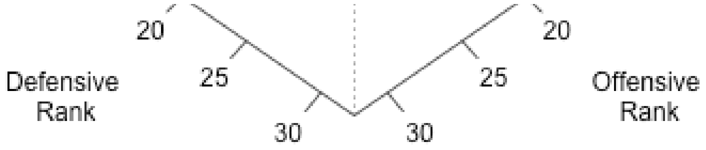




The following shows the probability of conversion of first down per offensive and defensive rankings with 3 yard to go.

In [43]:
```
%%R
z2 <- read.csv("z2.txt", header = F)
z2 <- as.matrix(z2)
x <- c(1:32); y <- c(1:32)
png("z2plot.png")
persp(x, y, z2, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) + 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 20, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First Down\n 3 Yards to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
dev.off()
persp(x, y, z2, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) + 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 20, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First Down\n 3 Yards to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
```

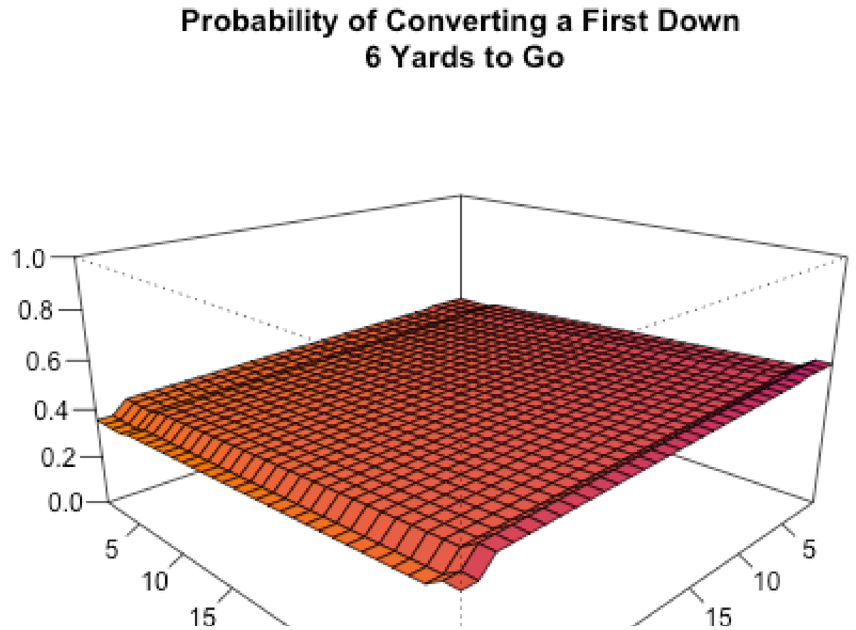


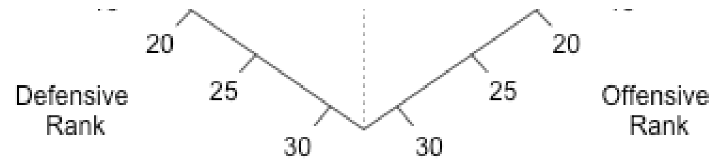


The following shows the probability of conversion of firt down per offensive and defensive rankings with 6 yard to go.

In [44]:

```
%%R
z3 <- read.csv("z3.txt", header = F)
z3 <- as.matrix(z3)
x <- c(1:32); y <- c(1:32)
png("z3plot.png")
persp(x, y, z3, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) + 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 20, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First Down\n 6 Yards to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
dev.off()
persp(x, y, z3, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) + 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 20, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First Down\n 6 Yards to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
```

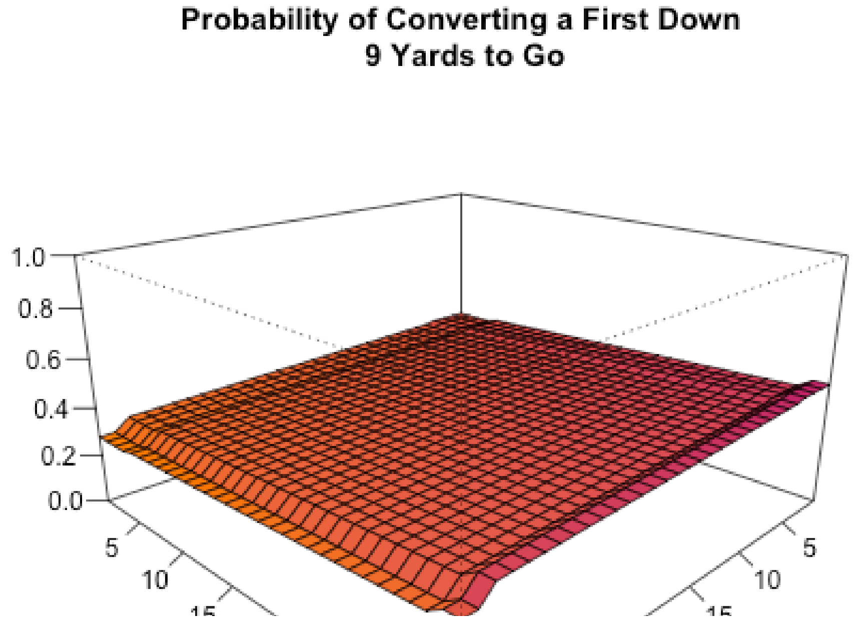


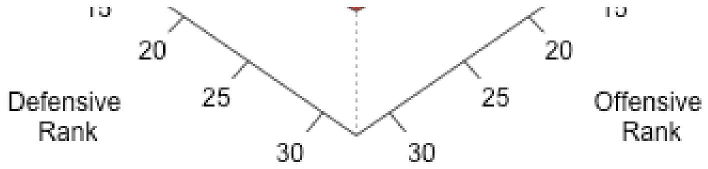


The following shows the probability of conversion of firt down per offensive and defensive rankings with 9 yard to go.

In [45]:

```
%%R
z4 <- read.csv("z4.txt", header = F)
z4 <- as.matrix(z4)
x <- c(1:32); y <- c(1:32)
png("z4plot.png")
persp(x, y, z4, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) +
 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 2
0, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First
 Down\n 9 Yards to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
dev.off()
persp(x, y, z4, expand = 0.5, zlim = range(0, 1), col = rgb(t(z1[-32, -32]) +
 0.2, (t(z1[-32, -32]))^2, z1[-32, -32] - 0.3), box = T, theta = 135, phi = 2
0, zlab = "", xlab = "", ylab = "", main = "Probability of Converting a First
 Down\n 9 Yards to Go", ticktype = "detailed")
text(-0.3,-0.35, "Defensive\n Rank")
text(0.3,-0.35, "Offensive\n Rank")
```





In [ ]: