# Intro to Data Science

## Project 2: Logistic Regression Analysis

**Chunwang Yuan, Rajendra Kumar Vechalapu, Saketh Yelamarthi**
**Fall 2021**

## Outline

1. Abstract
2. Theory
3. Exploratory Data analysis
4. Analysis Results & Explanation
5. Conclusion

## 1 Abstract

The main aim of this project is to perform logistic regression analysis on the given data set that represents whether a given e-mail is spam or not spam. The dataset contains 20 features that are used to determine whether an e-mail is spam or not spam. Before performing logistic regression, we perform feature elimination so that significant feature sets are used in model analysis. After modeling the data, we iterate the model for various threshold probability values and check the values of sensitivity and specificity for various thresholds.

Therefore, our goal is to find the optimal threshold value for which the true positive rate is close to 1 so that we build an optimum classification model that classifies a spam e-mail from ham.

## 2 Theory

Logistic Regression falls under Supervised Learning technique and is a type of regression that contains categorical response and one or multiple numerical and/or categorical predictors. It uses a generalized linear model to model a binart categorical variable using numerical and categorical predictors. Assuming the binomial distribution producing the outcome variable and therefore want to model p the probability of success for a given set of predictors.

There are various options but of all Logit function is used in this particular model.Logit function is defined as logarithm of p over (1-p) for p lies between 0 and 1.

In this particular dataset, we analyze email which are categorized into spam or not spam. This dataset uses 20 features to determine whether the given email is spam or not. We fit the model with the given data and plot the summary. From the summary table, we discard the features whose p-value lies below the significance level i.e., below 5%. By doing this, we are performing feature elimination. After performing feature elimination we are left with the significant features which are useful in determining the classification output.

After building the model, we iterate the model for various threshold probability values and check the values of True Positive, True Negative, False Positive and False Negative. From these values we calculate Sensitivity and Specificity. And we plot the graph of sensitivity and specificity.

From this, we find True positivity rate and False positivity rate and plot the graph of these values. The plotted curve should lie above the random case curve.

This plotted curve is called ROC curve-Receiver Operating Characteristic curve. From this curve, we find the optimal threshold probability value. We find this by determining the point on the ROC curve that is close to 1 i.e., the value of true positive rate is close to 1. The ideal point is that which has TPR value of 1 and FPR value of 0. So we choose the point which is close to the ideal point.

## 3 Exploratory Data analysis

**In this section, we created a logistic regression model to explorate the data.**

In [3]:

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from patsy import dmatrices
import matplotlib.pyplot as plt
import seaborn as sns
```

**Firstly, we load the data as the dataframe type set.**

In [35]:

```
df = pd.read_csv('DSCI6002_prj2_data.csv')
df.head()
```

Out[35]:

| | spam | to_multiple | from | cc | sent_email | time | image | attach | dollar | wi |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0 | 1 | 0 | 0 | 2011-12-31 22:16:41 | 0 | 0 | 0 | |
| **1** | 0 | 0 | 1 | 0 | 0 | 2011-12-31 23:03:59 | 0 | 0 | 0 | |
| **2** | 0 | 0 | 1 | 0 | 0 | 2012-01-01 08:00:32 | 0 | 0 | 4 | |
| **3** | 0 | 0 | 1 | 0 | 0 | 2012-01-01 01:09:49 | 0 | 0 | 0 | |
| **4** | 0 | 0 | 1 | 0 | 0 | 2012-01-01 02:00:01 | 0 | 0 | 0 | |

5 rows × 21 columns

In [5]:

```
df.shape
```

Out[5]:

```
(3921, 21)
```

In [6]:

```
df.dtypes
```

Out[6]:

```
spam              int64
to_multiple       int64
from              int64
cc                int64
sent_email        int64
time             object
image             int64
attach            int64
dollar            int64
winner           object
inherit           int64
viagra            int64
password          int64
num_char        float64
line_breaks       int64
format            int64
re_subj           int64
exclaim_subj      int64
urgent_subj       int64
exclaim_mess      int64
number           object
dtype: object
```

**Since the data type of the column of 'time' is object, we need to convert the value from object to numeric.**

In [41]:

```
df['time'] = pd.to_datetime(df['time'])
df['time'] = pd.to_numeric(df['time'])
```

**Because the title of 'from' column is one of keywords, we need change the title.**

In [42]:

```
df['from_'] = df['from']
```

In [43]:

```
df.head()
```

Out[43]:

| | spam | to_multiple | from | cc | sent_email | time | image | attac |
|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0 | 1 | 0 | 0 | 1325369801000000000 | 0 | |
| **1** | 0 | 0 | 1 | 0 | 0 | 1325372639000000000 | 0 | |
| **2** | 0 | 0 | 1 | 0 | 0 | 1325404832000000000 | 0 | |
| **3** | 0 | 0 | 1 | 0 | 0 | 1325380189000000000 | 0 | |
| **4** | 0 | 0 | 1 | 0 | 0 | 1325383201000000000 | 0 | |

5 rows × 22 columns

# General Model

### Data Preparation: using all the columns of dataset

In [44]:

```
y1 , X1 = dmatrices('spam~ to_multiple + cc + from_ + sent_email + time + \
                     image + attach + dollar + C(winner) + inherit + \
                     viagra + password + num_char + line_breaks + format + \
                     re_subj + exclaim_subj + urgent_subj + exclaim_mess + C(nu
X1
```

Out[44]:

| | Intercept | C(winner) [T.yes] | C(number) [T.none] | C(number) [T.small] | to_multiple | cc | from_ | sent |
|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | |
| **1** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | |
| **2** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | |
| **3** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | |
| **4** | 1.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 1.0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **3916** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | |
| **3917** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | |
| **3918** | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 0.0 | 1.0 | |
| **3919** | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 0.0 | 1.0 | |
| **3920** | 1.0 | 1.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | |

3921 rows × 22 columns

In [48]:

```
logit_res1 = sm.Logit(y1, X1).fit()
print(logit_res1.summary())
```

```
Warning: Maximum number of iterations has been exceeded.
         Current function value: 0.208584
         Iterations: 35
                           Logit Regression Results
==============================================================================
Dep. Variable:                   spam   No. Observations:
3921
Model:                          Logit   Df Residuals:
3899
Method:                           MLE   Df Model:
21
Date:                Fri, 03 Dec 2021   Pseudo R-squ.:
0.3288
Time:                        11:31:09   Log-Likelihood:
-817.86
converged:                      False   LL-Null:
-1218.6
Covariance Type:            nonrobust   LLR p-value:
4.446e-156
==============================================================================
                         coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
Intercept           -100.4727    151.207     -0.664      0.506
-396.834     195.888
C(winner)[T.yes]       2.0711      0.365      5.672      0.000
1.355       2.787
C(number)[T.none]      0.2953      0.220      1.345      0.179
-0.135       0.726
C(number)[T.small]    -0.8971      0.196     -4.576      0.000
-1.281      -0.513
to_multiple           -2.6802      0.327     -8.208      0.000
-3.320      -2.040
cc                     0.0188      0.022      0.854      0.393
-0.024       0.062
from_                -12.5392    146.412     -0.086      0.932
-299.501     274.423
sent_email           -22.8187   1801.354     -0.013      0.990
-3553.408    3507.771
time                8.472e-17   2.85e-17      2.974      0.003
2.89e-17    1.41e-16
image                 -1.7816      0.595     -2.996      0.003
-2.947      -0.616
attach                 0.7345      0.144      5.089      0.000
```

```
0.452       1.017
dollar               -0.0685      0.026     -2.588      0.010
-0.120      -0.017
inherit               0.3146      0.156      2.022      0.043
0.010       0.620
viagra                2.8520   3787.978      0.001      0.999
-7421.449    7427.153
password             -0.8545      0.297     -2.876      0.004
-1.437      -0.272
num_char              0.0506      0.024      2.127      0.033
0.004       0.097
line_breaks          -0.0055      0.001     -4.060      0.000
-0.008      -0.003
format               -0.6142      0.149     -4.136      0.000
-0.905      -0.323
re_subj              -1.6417      0.386     -4.248      0.000
-2.399      -0.884
exclaim_subj          0.1421      0.243      0.585      0.558
-0.334       0.618
urgent_subj           3.8845      1.317      2.950      0.003
1.303       6.466
exclaim_mess          0.0108      0.002      5.980      0.000
0.007       0.014
==============================================================
======================

Possibly complete quasi-separation: A fraction 0.28 of observati
ons can be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be ident
ified.
```

```
D:\program\Anaconda\lib\site-packages\statsmodels\base\model.py:
566: ConvergenceWarning: Maximum Likelihood optimization failed
to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

**Removing the columns (cc, from_, sent_email, viagra, exclaim_subj ) that have a zero beta with big probability.**

In [49]:

```
y2 , X2 = dmatrices('spam~ to_multiple + time + image + attach + dollar + C(wi
                     password + num_char + line_breaks + format + \
                     re_subj + urgent_subj + exclaim_mess + C(number)', df , re
X2
```

Out[49]:

| | Intercept | C(winner) [T.yes] | C(number) [T.none] | C(number) [T.small] | to_multiple | time | in |
|---|---|---|---|---|---|---|---|
| **0** | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.325370e+18 | |
| **1** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 1.325373e+18 | |
| **2** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 1.325405e+18 | |
| **3** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 1.325380e+18 | |
| **4** | 1.0 | 0.0 | 1.0 | 0.0 | 0.0 | 1.325383e+18 | |
| **...** | ... | ... | ... | ... | ... | ... | |
| **3916** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 1.333127e+18 | |
| **3917** | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 1.333178e+18 | |
| **3918** | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 1.333099e+18 | |
| **3919** | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 1.332925e+18 | |
| **3920** | 1.0 | 1.0 | 0.0 | 1.0 | 0.0 | 1.333160e+18 | |

3921 rows × 17 columns

In [50]:

```
logit_res2 = sm.Logit(y2, X2).fit()
print(logit_res2.summary())
```

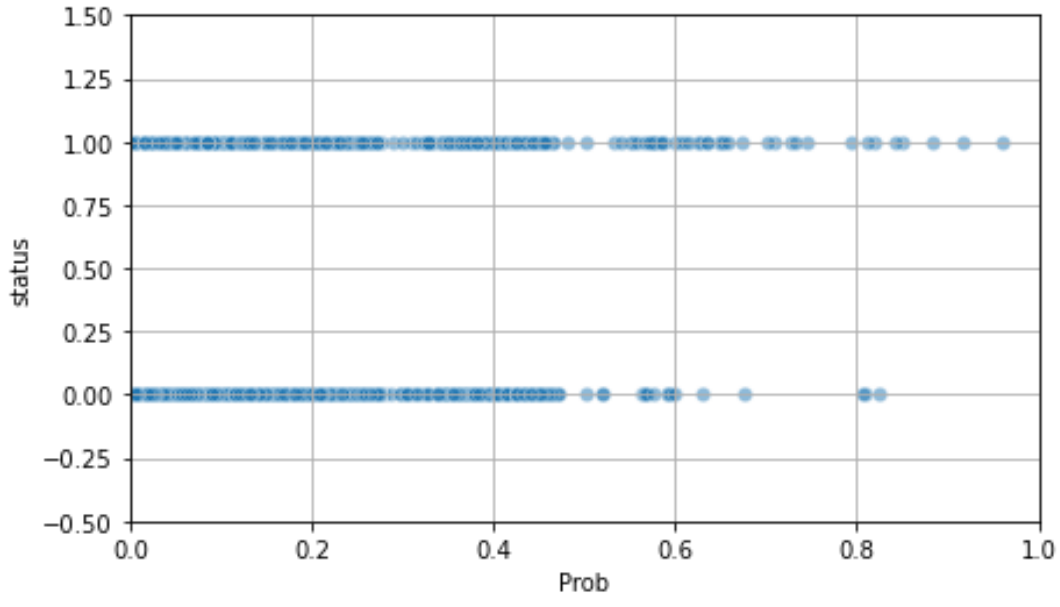
```
Optimization terminated successfully.
         Current function value: 0.228731
         Iterations 10
                           Logit Regression Results
==============================================================================
Dep. Variable:                   spam   No. Observations:
3921
Model:                          Logit   Df Residuals:
3904
Method:                           MLE   Df Model:
16
Date:                Fri, 03 Dec 2021   Pseudo R-squ.:
0.2640
Time:                        11:31:11   Log-Likelihood:
-896.85
converged:                       True   LL-Null:
-1218.6
Covariance Type:            nonrobust   LLR p-value:
1.353e-126
==============================================================================
                       coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
Intercept            -94.0394     36.311     -2.590      0.010
-165.208     -22.871
C(winner)[T.yes]       2.1489      0.361      5.949      0.000
1.441       2.857
C(number)[T.none]     -0.0752      0.211     -0.357      0.721
-0.488       0.338
C(number)[T.small]    -0.8818      0.192     -4.594      0.000
-1.258      -0.506
to_multiple           -2.6580      0.314     -8.460      0.000
-3.274      -2.042
time                7.049e-17   2.73e-17      2.581      0.010
1.7e-17    1.24e-16
image                 -2.1348      0.936     -2.281      0.023
-3.969      -0.301
attach                 0.4826      0.099      4.894      0.000
0.289       0.676
dollar                -0.0645      0.025     -2.542      0.011
-0.114      -0.015
inherit                0.3530      0.155      2.275      0.023
0.049       0.657
password              -0.8272      0.300     -2.756      0.006
```

```
-1.415      -0.239
num_char              0.0625      0.022      2.853      0.004
0.020       0.106
line_breaks          -0.0057      0.001     -4.438      0.000
-0.008      -0.003
format               -0.8599      0.138     -6.209      0.000
-1.131      -0.588
re_subj              -2.9963      0.386     -7.770      0.000
-3.752      -2.241
urgent_subj           3.7512      1.063      3.530      0.000
1.668       5.834
exclaim_mess          0.0098      0.002      5.629      0.000
0.006       0.013
==================================================================
======================
```

# Thresholds

In [51]:

```
y_pred = logit_res.predict(X2)
df_pred = pd.DataFrame({'Prob': y_pred,
                        'status': df.spam,
                        })
plt.figure(figsize = (7,4))
sns.scatterplot(data = df_pred, x = 'Prob', y = 'status',
                alpha = 0.5, s = 40)
plt.ylim([-.5, 1.5]); plt.xlim([0,1]);
plt.grid()
```




# ROC curve

In [52]:

```
#calculate the tpr and fpr
def calculate_tpr_fpr(df, threshold):
    TP = df_pred[(df_pred.status == 1) & (df_pred.Prob >= threshold)].count()[
    FN = df_pred[(df_pred.status == 1) & (df_pred.Prob < threshold)].count()[0
    TN = df_pred[(df_pred.status == 0) & (df_pred.Prob < threshold)].count()[0
    FP = df_pred[(df_pred.status == 0) & (df_pred.Prob >= threshold)].count()[

    sensitivity = TP/(TP + FN)
    specificity = TN/(FP + TN)
    tpr = sensitivity
    fpr = 1- specificity
    return tpr, fpr

tpr_n = []
fpr_n = []
thresholds = [i/100.0 for i in range(0,100)]
for threshold in thresholds:
    tpr, fpr = calculate_tpr_fpr(df, threshold)
    tpr_n.append(tpr)
    fpr_n.append(fpr)
```

In [53]:

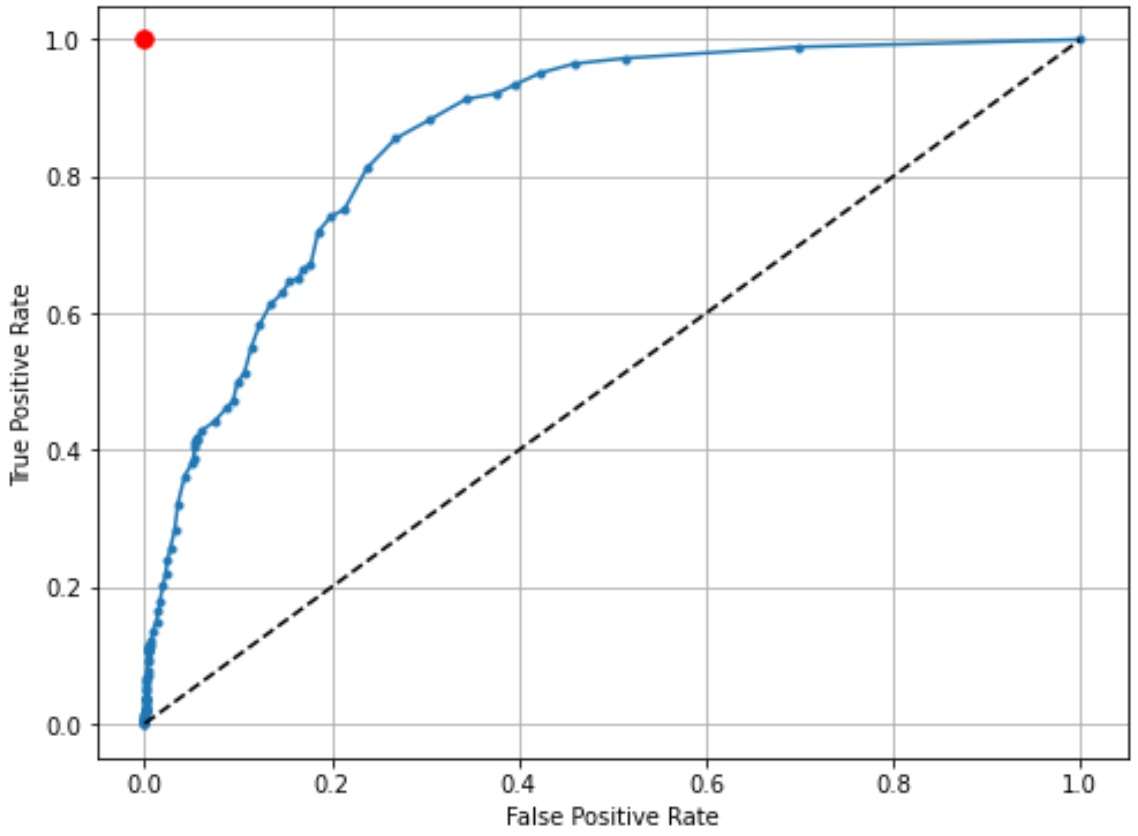
```
tpr_n = np.flip(tpr_n)
fpr_n = np.flip(fpr_n)
```

In [54]:

```
auc = np.trapz(tpr_n, fpr_n)
print("AUC: {}".format(auc))

plt.figure(figsize = (8,6))
plt.plot(fpr_n, tpr_n, '.-')
plt.plot([0,1], [0,1], 'k--')
plt.plot([0], [1], 'r.', markersize = 15)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.grid()
```

```
AUC: 0.8605888287978851
```




## Optimal point in ROC curve

In [55]:

```
import math

# shortest distance between best point and point on curve
best_point= (.0, 1.0)
distances = []
for i in zip(fpr_n, tpr_n):
    distance = math.dist(best_point, i)
    distances.append(distance)

min_idx = np.argmin(distances)
print("index of min:\t{}\n"
         "threshold:\t{:2}\n"
         "fpr:\t\t{:.2}\n"
         "tpr:\t\t{:.2}".format(
         min_idx,
         thresholds[99-min_idx],
         fpr_n[min_idx],
         tpr_n[min_idx
              ]))
```

```
index of min:   89
threshold:      0.1
fpr:            0.24
tpr:            0.81
```

## 4 Analysis Results & Explanation

**RESULTS**:

In [56]:

```
print(logit_res1.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                   spam   No. Observations:
3921
Model:                          Logit   Df Residuals:
3899
Method:                           MLE   Df Model:
21
Date:                Fri, 03 Dec 2021   Pseudo R-squ.:
0.3288
Time:                        11:32:28   Log-Likelihood:
-817.86
converged:                      False   LL-Null:
-1218.6
Covariance Type:            nonrobust   LLR p-value:
4.446e-156
==============================================================================
                         coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
Intercept           -100.4727    151.207     -0.664      0.506
-396.834     195.888
C(winner)[T.yes]       2.0711      0.365      5.672      0.000
1.355       2.787
C(number)[T.none]      0.2953      0.220      1.345      0.179
-0.135       0.726
C(number)[T.small]    -0.8971      0.196     -4.576      0.000
-1.281      -0.513
to_multiple           -2.6802      0.327     -8.208      0.000
-3.320      -2.040
cc                     0.0188      0.022      0.854      0.393
-0.024       0.062
from_                -12.5392    146.412     -0.086      0.932
-299.501     274.423
sent_email           -22.8187   1801.354     -0.013      0.990
-3553.408    3507.771
time                8.472e-17   2.85e-17      2.974      0.003
2.89e-17    1.41e-16
image                 -1.7816      0.595     -2.996      0.003
-2.947      -0.616
attach                 0.7345      0.144      5.089      0.000
0.452       1.017
dollar                -0.0685      0.026     -2.588      0.010
-0.120      -0.017
inherit                0.3146      0.156      2.022      0.043
```

```
0.010       0.620
viagra                2.8520   3787.978      0.001      0.999
-7421.449    7427.153
password             -0.8545      0.297     -2.876      0.004
-1.437      -0.272
num_char              0.0506      0.024      2.127      0.033
0.004       0.097
line_breaks          -0.0055      0.001     -4.060      0.000
-0.008      -0.003
format               -0.6142      0.149     -4.136      0.000
-0.905      -0.323
re_subj              -1.6417      0.386     -4.248      0.000
-2.399      -0.884
exclaim_subj          0.1421      0.243      0.585      0.558
-0.334       0.618
urgent_subj           3.8845      1.317      2.950      0.003
1.303       6.466
exclaim_mess          0.0108      0.002      5.980      0.000
0.007       0.014
==============================================================
======================

Possibly complete quasi-separation: A fraction 0.28 of observati
ons can be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be ident
ified.
```

In [57]:

```
print(logit_res2.summary())
```

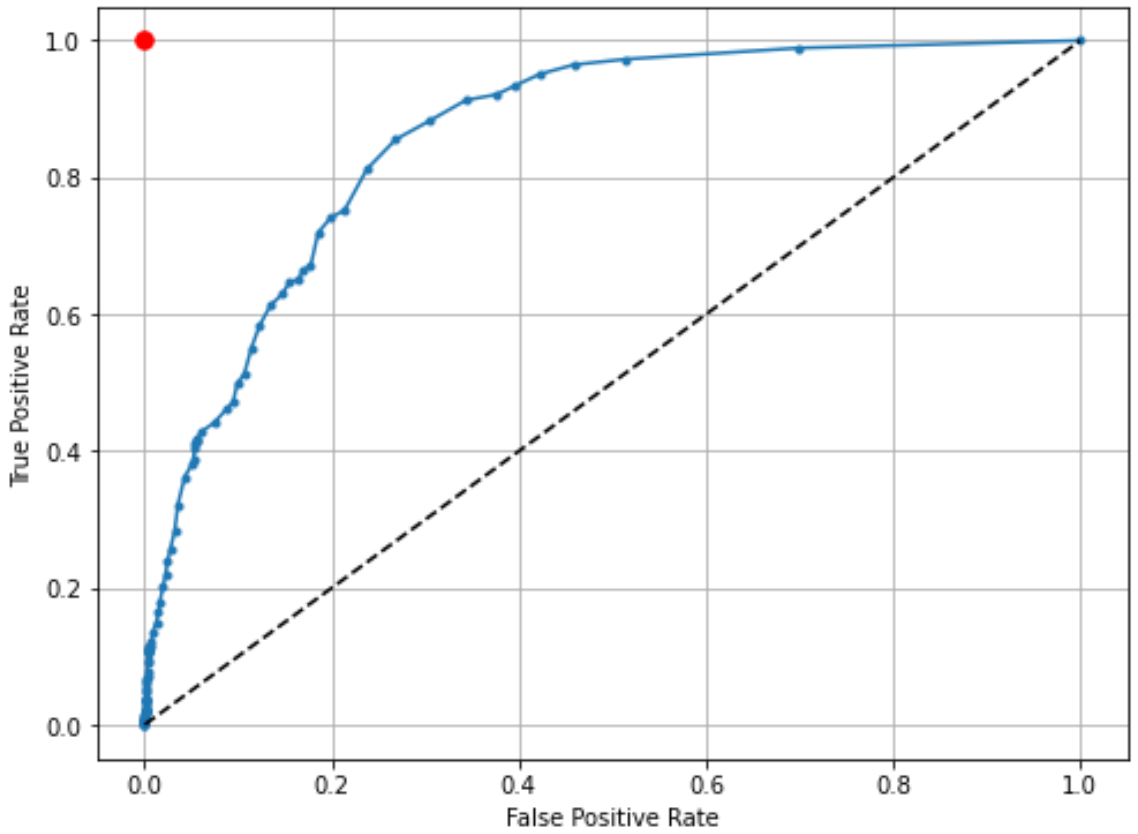
```
                           Logit Regression Results
==============================================================================
Dep. Variable:                   spam   No. Observations:
3921
Model:                          Logit   Df Residuals:
3904
Method:                           MLE   Df Model:
16
Date:                Fri, 03 Dec 2021   Pseudo R-squ.:
0.2640
Time:                        11:32:29   Log-Likelihood:
-896.85
converged:                       True   LL-Null:
-1218.6
Covariance Type:            nonrobust   LLR p-value:
1.353e-126
==============================================================================
                         coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
Intercept            -94.0394     36.311     -2.590      0.010
-165.208     -22.871
C(winner)[T.yes]       2.1489      0.361      5.949      0.000
1.441       2.857
C(number)[T.none]     -0.0752      0.211     -0.357      0.721
-0.488       0.338
C(number)[T.small]    -0.8818      0.192     -4.594      0.000
-1.258      -0.506
to_multiple           -2.6580      0.314     -8.460      0.000
-3.274      -2.042
time                7.049e-17   2.73e-17      2.581      0.010
1.7e-17    1.24e-16
image                 -2.1348      0.936     -2.281      0.023
-3.969      -0.301
attach                 0.4826      0.099      4.894      0.000
0.289       0.676
dollar                -0.0645      0.025     -2.542      0.011
-0.114      -0.015
inherit                0.3530      0.155      2.275      0.023
0.049       0.657
password              -0.8272      0.300     -2.756      0.006
-1.415      -0.239
num_char               0.0625      0.022      2.853      0.004
0.020       0.106
line_breaks           -0.0057      0.001     -4.438      0.000
```

```
-0.008      -0.003
format                -0.8599      0.138      -6.209      0.000
-1.131      -0.588
re_subj               -2.9963      0.386      -7.770      0.000
-3.752      -2.241
urgent_subj            3.7512      1.063       3.530      0.000
1.668       5.834
exclaim_mess           0.0098      0.002       5.629      0.000
0.006       0.013
==============================================================
======================
```

In [60]:

```
print("AUC: {}".format(auc))
plt.figure(figsize = (8,6))
plt.plot(fpr_n, tpr_n, '.-')
plt.plot([0,1], [0,1], 'k--')
plt.plot([0], [1], 'r.', markersize = 15)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.grid()
```

```
AUC: 0.8605888287978851
```



In [58]:

```
print("index of min:\t{}\n"
        "optimal threshold:\t{:2}\n"
        "fpr:\t\t{:.2}\n"
        "tpr:\t\t{:.2}".format(
        min_idx,
        thresholds[99-min_idx],
        fpr_n[min_idx],
        tpr_n[min_idx
             ]))
```

```
index of min:   89
optimal threshold:      0.1
fpr:            0.24
tpr:            0.81
```

**ANALYSIS & EXPLANATION:**

1. At the beginning of this analysis with logistic regression method, we applyed all the columns to our first logistic regression model to detect the columns that have zero coefficient in our model. From the details of logit_res1.summary(), it given us a specifical result that P-values of five columns of cc, from_, sent_email, viagra and exclaim_subj were far more than alpha(0.05) and P-values of rest columns was less than alpha. For those P-values were significantly more than 0.05, we can't reject Null Hypothesis and concluded that the coefficient values of those columns were 0.
2. Since we concluded the the coefficient values of five columns of cc, from_, sent_email, viagra and exclaim_subj were 0, we removed those columns and applyed rest of columns into our second model to make a further detection. The informations of logit_res2.summary() revealed the P-value of remained columns were less than 0.05. We rejected the Null Hypothesis and concluded that the coefficient values of those columns were not 0.
3. Based on the Roc curve, the second logistic regression model got AUC of 0.86. Such high AUC illustrated that the model is splendid and accurate for the dataset.
4. Using the method to find the shortest distance between best point and point on curve, the optimal point was (fpr: 0.24, tpr: 0.81) corresponding to the optimal threshold of 0.1.

# 5 Conclusion

**According to the results of logistic regression analysis, we can conclude the five columns of cc, from_, sent_email, viagra and exclaim_subj didn't relate to the target column.**

**By applying the rest columns into our model, we can predict the email is spam(1) or not(0) with a accuracy of 0.81 at 1 and a accuracy of 0.76 at 0.**