In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import pandas_profiling as pp
```

In [365]:
```
assessments=pd.read_csv('assessments.csv')
assessments.head(5)
```

Out[365]:

	code_module	code_presentation	id_assessment	assessment_type	date	weight
0	AAA	2013J	1752	TMA	19	10.0
1	AAA	2013J	1753	TMA	54	20.0
2	AAA	2013J	1754	TMA	117	20.0
3	AAA	2013J	1755	TMA	166	20.0
4	AAA	2013J	1756	TMA	215	30.0

In [3]:
```
assessments[['code_module','code_presentation']].duplicated().sum()
```

Out[3]: 184

In [4]:
```
pp.ProfileReport(assessments)
```

Out[4]:

Overview

Dataset info

Number of variables	6
Number of observations	206
Total Missing (%)	0.0%
Total size in memory	9.7 KiB
Average record size in memory	48.4 B

Variables types

Numeric	2
Categorical	4
Boolean	0

In [4]:
```
courses=pd.read_csv('courses.csv')
```

In [6]:
```
pp.ProfileReport(courses)
```

Out[6]:

Overview

Dataset info

Number of variables	3
Number of observations	22
Total Missing (%)	0.0%
Total size in memory	608.0 B
Average record size in memory	27.6 B

Variables types

Numeric	1
Categorical	2
Boolean	0
Date	0
Text (Unique)	0

In [5]:
```
studentAssessment=pd.read_csv('studentAssessment.csv')
studentAssessment.head()
```

Out[5]:

	id_assessment	id_student	date_submitted	is_banked	score
0	1752	11391	18	0	78
1	1752	28400	22	0	70
2	1752	31604	17	0	72
3	1752	32885	26	0	69
4	1752	38053	19	0	79

In [6]:
```
studentAssessment[['id_assessment']].duplicated().sum()
```

Out[6]: 173724

In [7]:
```
# np.array(studentAssessment['score']).astype(str).astype(int)

missingvallist=np.where(pd.to_numeric(studentAssessment['score'].astype(str).
```

In [8]:
```
missingstudentscorelistofid=list(studentAssessment.iloc[missingvallist]['id_s
# missingstudentscorelistofid
```

In [11]:
```
pp.ProfileReport(studentAssessment)
```

Out[11]:

Overview

Dataset info

Number of variables	5
Number of observations	173912
Total Missing (%)	0.0%
Total size in memory	6.6 MiB
Average record size in memory	40.0 B

Variables types

Numeric	3
Categorical	1
Boolean	1
Date	0
Text (Unique)	0

In [9]:
```
studentInfo=pd.read_csv('studentInfo.csv')
studentInfo.head()
```

Out[9]:

	code_module	code_presentation	id_student	gender	region	highest_education	imd_ban
0	AAA	2013J	11391	M	East Anglian Region	HE Qualification	90-100%
1	AAA	2013J	28400	F	Scotland	HE Qualification	20-30%
2	AAA	2013J	30268	F	North Western Region	A Level or Equivalent	30-40%
3	AAA	2013J	31604	F	South East Region	A Level or Equivalent	50-60%
4	AAA	2013J	32885	F	West Midlands Region	Lower Than A Level	50-60%

In [10]:
```
studentInfo['final_result'].value_counts()
```

Out[10]:
```
Pass           12361
Withdrawn      10156
Fail            7052
Distinction     3024
Name: final_result, dtype: int64
```

In [11]:
```
studentInfo.iloc[list((studentInfo[studentInfo['id_student'].isin(missingstuc
```

Out[11]:

	code_module	code_presentation	id_student	gender	region	highest_education	im
108	AAA	2013J	260355	F	London Region	A Level or Equivalent	
227	AAA	2013J	721259	F	South Region	Lower Than A Level	
466	AAA	2014J	260355	F	London Region	A Level or Equivalent	
638	AAA	2014J	721259	F	South Region	Lower Than A Level	
733	AAA	2014J	2606802	M	North Region	A Level or Equivalent	
753	BBB	2013B	33666	F	London Region	A Level or Equivalent	
843	BBB	2013B	186780	F	North Western	A Level or	

In [15]:
```
pp.ProfileReport(studentInfo)
```

Out[15]:

Overview

Dataset info

Number of variables	12
Number of observations	32593
Total Missing (%)	0.0%
Total size in memory	3.0 MiB
Average record size in memory	96.0 B

Variables types

Numeric	3
Categorical	9
Boolean	0

In [13]:
```
studentRegistration=pd.read_csv('studentRegistration.csv')
studentRegistration.head()
```

Out[13]:

	code_module	code_presentation	id_student	date_registration	date_unregistration
0	AAA	2013J	11391	-159	?
1	AAA	2013J	28400	-53	?
2	AAA	2013J	30268	-92	12
3	AAA	2013J	31604	-52	?
4	AAA	2013J	32885	-176	?

In [17]:
```
pp.ProfileReport(studentRegistration)
```

Out[17]:

Overview

Dataset info

Number of variables	5
Number of observations	32593
Total Missing (%)	0.0%
Total size in memory	1.2 MiB
Average record size in memory	40.0 B

Variables types

Numeric	1
Categorical	4
Boolean	0
Date	0

In [14]:
```
studentVle=pd.read_csv('studentVle.csv')
studentVle.head()
```

Out[14]:

	code_module	code_presentation	id_student	id_site	date	sum_click
0	AAA	2013J	28400	546652	-10	4
1	AAA	2013J	28400	546652	-10	1
2	AAA	2013J	28400	546652	-10	1
3	AAA	2013J	28400	546614	-10	11
4	AAA	2013J	28400	546714	-10	1

In [19]:
```
pp.ProfileReport(studentVle)
```

Out[19]:

Overview

Dataset info

Number of variables	6
Number of observations	10655280
Total Missing (%)	0.0%
Total size in memory	487.8 MiB
Average record size in memory	48.0 B

Variables types

Numeric	4
Categorical	2
Boolean	0
Date	0
Text (Unique)	0

In [15]:
```
vle=pd.read_csv('vle.csv')
vle.head()
```

Out[15]:

	id_site	code_module	code_presentation	activity_type	week_from	week_to
0	546943	AAA	2013J	resource	?	?
1	546712	AAA	2013J	oucontent	?	?
2	546998	AAA	2013J	resource	?	?
3	546888	AAA	2013J	url	?	?
4	547035	AAA	2013J	resource	?	?

In [16]:
```
vle.activity_type.value_counts()
```

Out[16]:
```
resource          2660
subpage           1055
oucontent          996
url                886
forumng            194
quiz               127
page               102
oucollaborate       82
questionnaire       61
ouwiki              49
dataplus            28
externalquiz        26
homepage            22
glossary            21
ouelluminate        21
dualpane            20
repeatactivity       5
htmlactivity         4
sharedsubpage        3
folder               2
Name: activity_type, dtype: int64
```

In [22]:
```
pp.ProfileReport(vle)
```

Unique (%) 0.1%
Missing (%) 0.0%
Missing (n) 0

FFF	1967
DDD	1708
BBB	1154
Other values (4)	1535

Toggle details

code_presentation

Categorical

Distinct count	4
Unique (%)	0.1%
Missing (%)	0.0%
Missing (n)	0

In [17]:
```
vle.head(14)
```

Out[17]:

	id_site	code_module	code_presentation	activity_type	week_from	week_to
0	546943	AAA	2013J	resource	?	?
1	546712	AAA	2013J	oucontent	?	?
2	546998	AAA	2013J	resource	?	?
3	546888	AAA	2013J	url	?	?
4	547035	AAA	2013J	resource	?	?
5	546614	AAA	2013J	homepage	?	?
6	546897	AAA	2013J	url	?	?
7	546678	AAA	2013J	oucontent	?	?
8	546933	AAA	2013J	resource	?	?
9	546708	AAA	2013J	oucontent	?	?
10	546995	AAA	2013J	resource	?	?
11	546884	AAA	2013J	url	?	?
12	547031	AAA	2013J	resource	?	?
13	546891	AAA	2013J	url	?	?

In [368]:
```
vle.activity_type.value_counts()
```

Out[368]:
```
resource          2660
subpage           1055
oucontent          996
url                886
forumng            194
quiz               127
page               102
oucollaborate       82
questionnaire       61
ouwiki              49
dataplus            28
externalquiz        26
homepage            22
glossary            21
ouelluminate        21
dualpane            20
repeatactivity       5
htmlactivity         4
sharedsubpage        3
folder               2
Name: activity_type, dtype: int64
```

In [18]:
```
studentInfo[(studentInfo['code_module']=='BBB') & (studentInfo['code_presenta
```

					Region	Equivalent
4741	BBB	2013J	2650236	F	South East Region	Lower Than A Level
4742	BBB	2013J	2656860	F	East Anglian Region	Lower Than A Level
4743	BBB	2013J	2657960	F	North Western Region	Lower Than A Level
4744	BBB	2013J	2662716	F	East Anglian Region	Lower Than A Level
4745	BBB	2013J	2664036	F	London Region	A Level or Equivalent
4746	BBB	2013J	2680344	F	Scotland	HE Qualification
					South	

In [19]:
```
student = pd.merge(studentInfo, courses,  how='left', on=['code_module','code
```

In [20]:
```
student = pd.merge(student, studentRegistration,  how='left', on=['code_modul
student.head()
```

Out[20]:

ation	imd_band	age_band	num_of_prev_attempts	studied_credits	disability	final_result	modu
ation	90-100%	55<=	0	240	N	Pass	
ation	20-30%	35-55	0	60	N	Pass	
vel or alent	30-40%	35-55	0	60	Y	Withdrawn	
vel or alent	50-60%	35-55	0	60	N	Pass	
Level	50-60%	0-35	0	60	N	Pass	

In [21]:
```
vle["week_from"]= vle["week_from"].str.replace("?",'-999')
vle["week_to"]= vle["week_to"].str.replace("?",'-999')
vle.head()
```

Out[21]:

	id_site	code_module	code_presentation	activity_type	week_from	week_to
0	546943	AAA	2013J	resource	-999	-999
1	546712	AAA	2013J	oucontent	-999	-999
2	546998	AAA	2013J	resource	-999	-999
3	546888	AAA	2013J	url	-999	-999
4	547035	AAA	2013J	resource	-999	-999

In [22]:
```
temp1=studentVle.groupby(['id_student', 'code_module', 'code_presentation']).
temp1.columns=['Sum_id_site','Sum_date','Sum_sum_click']
temp1.drop('Sum_id_site',1,inplace=True)

temp2=studentVle.groupby(['id_student', 'code_module', 'code_presentation']).
temp2.columns=['Mean_id_site','Mean_date','Mean_sum_click']
temp2.drop('Mean_id_site',1,inplace=True)

temp3=studentVle.groupby(['id_student', 'code_module', 'code_presentation']).
temp3.columns=['Count_id_site','Count_date','Count_sum_click']
temp3.drop(['Count_id_site','Count_date'],1,inplace=True)

temp3.columns=['Count_ALL_Click']
temp4 = studentVle.groupby(['id_student', 'code_module', 'code_presentation']
temp4.columns=['Mode_id_site','Mode_date','Mode_sum_click']
temp4.head()

temp5=pd.concat([temp1,temp2,temp3,temp4], axis=1)
```

In [23]:
```
temp5.reset_index(level=['id_student','code_module','code_presentation'],inpl
print("temp5 shape is :",temp5.shape)
temp5.head()
```

```
temp5 shape is : (29228, 11)
```

Out[23]:

id_student	code_module	code_presentation	Sum_date	Sum_sum_click	Mean_date	Mean_sum
6516	AAA	2014J	73140	2791	110.483384	4.2
8462	DDD	2013J	11247	646	37.490000	2.1
8462	DDD	2014J	40	10	10.000000	2.5
11391	AAA	2013J	20018	934	102.132653	4.7
23629	BBB	2013B	2539	161	43.033898	2.7

In [24]:
```
studentVle.shape
```

Out[24]: (10655280, 6)

In [25]:
```
student2 = pd.merge(studentVle, vle,  how='left',on=['id_site','code_module',
student2.head()
```

Out[25]:

	code_module	code_presentation	id_student	id_site	date	sum_click	activity_type	week_
0	AAA	2013J	28400	546652	-10	4	forumng	
1	AAA	2013J	28400	546652	-10	1	forumng	
2	AAA	2013J	28400	546652	-10	1	forumng	
3	AAA	2013J	28400	546614	-10	11	homepage	
4	AAA	2013J	28400	546714	-10	1	oucontent	

In [26]:
```
one_hot = pd.get_dummies(student2['activity_type'],prefix='ActivityType')
student2 = student2.drop('activity_type',axis = 1)
student2 = student2.join(one_hot)
student2.head()
student2.drop(['ActivityType_sharedsubpage','ActivityType_repeatactivity'],1,
student2.head()
```

Out[26]:

	ActivityType_dataplus	ActivityType_dualpane	...	ActivityType_oucollaborate	ActivityType_oucor
	0	0	...	0	
	0	0	...	0	
	0	0	...	0	
	0	0	...	0	
	0	0	...	0	

In [27]:

```
listofactivity=['ActivityType_dataplus','ActivityType_dualpane','ActivityType
temp1=student2.groupby(['id_student', 'code_module', 'code_presentation']).su
listofactivity2=[x+'_Sum' for x in listofactivity]
temp1.columns=listofactivity2

temp2=student2.groupby(['id_student', 'code_module', 'code_presentation']).me
listofactivity2=[x+'_Mean' for x in listofactivity]
temp2.columns=listofactivity2

temp3=student2.groupby(['id_student', 'code_module', 'code_presentation']).co
listofactivity2=[x+'_Count' for x in listofactivity]
temp3.columns=listofactivity2

dummystudent2=student2.copy()
dummystudent2.drop(['id_site','date','sum_click','week_from','week_to'],axis=
temp4=dummystudent2.groupby(['id_student', 'code_module', 'code_presentation'
listofactivity2=[x+'_Mode' for x in listofactivity]
temp4.columns=listofactivity2
temp4.ix[:,'ActivityType_dataplus':].sum()
temp5=pd.concat([temp1,temp2,temp3,temp4], axis=1)
temp5.head()

temp1=studentVle.groupby(['id_student', 'code_module', 'code_presentation']).
temp1.columns=['Sum_id_site','Sum_date','Sum_sum_click']
temp1.drop('Sum_id_site',1,inplace=True)

temp2=studentVle.groupby(['id_student', 'code_module', 'code_presentation']).
temp2.columns=['Mean_id_site','Mean_date','Mean_sum_click']
temp2.drop('Mean_id_site',1,inplace=True)

temp3=studentVle.groupby(['id_student', 'code_module', 'code_presentation']).
temp3.columns=['Count_id_site','Count_date','Count_sum_click']
temp3.drop(['Count_id_site','Count_date'],1,inplace=True)

temp3.columns=['Count_ALL_Click']
temp4 = studentVle.groupby(['id_student', 'code_module', 'code_presentation']
temp4.columns=['Mode_id_site','Mode_date','Mode_sum_click']
temp4.head()

listofweek=['week_from','week_to']
temp7=student2.groupby(['id_student', 'code_module', 'code_presentation']).ma
listofweek=[x+'_Max' for x in listofweek]
temp7.columns=listofweek


temp6=pd.concat([temp5,temp1,temp2,temp3,temp4,temp7], axis=1)

temp6.reset_index(level=['id_student','code_module','code_presentation'],inpl
print("temp6 shape is :",temp6.shape)
studentVle_vle=temp6.copy()
studentVle_vle.head()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:19: Deprec
ationWarning:
.ix is deprecated. Please use
```

```
.loc for label based indexing or
.iloc for positional indexing

See the documentation here:
http://pandas.pydata.org/pandas-docs/stable/indexing.html#ix-indexer-is-deprecated (http://pandas.pydata.org/pandas-docs/stable/indexing.html#ix-indexer-is-deprecated)

temp6 shape is : (29228, 85)
```

Out[27]:

	id_student	code_module	code_presentation	ActivityType_dataplus_Sum	ActivityType_dualpa
0	6516	AAA	2014J	4.0	
1	8462	DDD	2013J	0.0	
2	8462	DDD	2014J	0.0	
3	11391	AAA	2013J	0.0	
4	23629	BBB	2013B	0.0	

5 rows × 85 columns

In [28]:
```
studentAssessment_assesment = pd.merge(studentAssessment, assessments,  how='
```

In [29]:
```
studentAssessment_assesment['date'].astype(str).str.replace('?','-99').astype
126.82313468880814 +(2865*99)/(173912-2865)
# 2865 are ? and we are replacing them by -99 which leads to actual mean of 1
studentAssessment_assesment['date'].astype(str).str.replace('?','128').astype
studentAssessment_assesment['date']=studentAssessment_assesment['date'].astyp
```

In [30]:
```
studentAssessment_assesment['score'].astype(str).str.replace('?','-99').astyp
75.62569000414003 +(173*99)/(173912-173)
# studentAssessment_assesment['score'].astype(str).str.replace('?','75').asty
studentAssessment_assesment['score']=studentAssessment_assesment['score'].ast
studentAssessment_assesment['score'].mean()
```

Out[30]: 75.79877754266525

In [31]:
```
one_hot = pd.get_dummies(studentAssessment_assesment['assessment_type'],prefi
studentAssessment_assesment = studentAssessment_assesment.drop('assessment_ty
studentAssessment_assesment = studentAssessment_assesment.join(one_hot)
studentAssessment_assesment.head()
```

Out[31]:

	id_assessment	id_student	date_submitted	is_banked	score	code_module	code_presentat
0	1752	11391	18	0	78	AAA	20
1	1752	28400	22	0	70	AAA	20
2	1752	31604	17	0	72	AAA	20
3	1752	32885	26	0	69	AAA	20
4	1752	38053	19	0	79	AAA	20

In [32]:
```
cols=['id_assessment','date_submitted','is_banked','score','date','weight','a
temp1=studentAssessment_assesment.groupby(['id_student','code_module','code_p
cols2=[x+'_Sum' for x in cols]
temp1.columns=cols2

temp2=studentAssessment_assesment.groupby(['id_student','code_module','code_p
cols2=[x+'_Mean' for x in cols]
temp2.columns=cols2

temp3=studentAssessment_assesment.groupby(['id_student','code_module','code_p
cols2=[x+'_Max' for x in cols]
temp3.columns=cols2

temp4=studentAssessment_assesment.groupby(['id_student','code_module','code_p
cols2=[x+'_Min' for x in cols]
temp4.columns=cols2

temp5=studentAssessment_assesment.groupby(['id_student','code_module','code_p
cols2=[x+'_Count' for x in cols]
temp5.columns=cols2

temp6=studentAssessment_assesment.groupby(['id_student','code_module','code_p
cols2=[x+'_Mode' for x in cols]
temp6.columns=cols2
```

In [33]:
```
temp7=pd.concat([temp1,temp2,temp3,temp4,temp5,temp6], axis=1)
temp7.head()
```

Out[33]:

			id_assessment_Sum	date_submitted_Sum	is_ban
id_student	**code_module**	**code_presentation**			
6516	**AAA**	**2014J**	8800	558	
8462	**DDD**	**2013J**	76047	165	
		2014J	101454	-4	
11391	**AAA**	**2013J**	8770	562	
23629	**BBB**	**2013B**	59952	223	

5 rows × 54 columns

In [34]:
```
temp7.reset_index(level=['id_student','code_module','code_presentation'],inpl
temp7.shape
```

Out[34]: (25843, 57)

In [35]:
```
studentAssessment_assesment_new=temp7.copy()
studentAssessment_assesment_new[['id_student','code_module','code_presentatic
```

Out[35]: 0

In [370]:
```
studentAssessment_assesment_new.head()
```

Out[370]:

	id_student	code_module	code_presentation	id_assessment_Sum	date_submitted_Sum	is_
0	6516	AAA	2014J	8800	558	
1	8462	DDD	2013J	76047	165	
2	8462	DDD	2014J	101454	-4	
3	11391	AAA	2013J	8770	562	
4	23629	BBB	2013B	59952	223	

5 rows × 57 columns

In [369]:
```
pp.ProfileReport(studentAssessment_assesment_new)
```

Out[369]:

Overview

Dataset info

Number of variables	57
Number of observations	25843
Total Missing (%)	0.0%
Total size in memory	8.4 MiB
Average record size in memory	340.0 B

Variables types

Numeric	25
Categorical	2
Boolean	9
Date	0

In [36]:
```
student.shape
```

Out[36]: (32593, 15)

In [37]:
```
studentVle_vle.head()
```

Out[37]:

	id_student	code_module	code_presentation	ActivityType_dataplus_Sum	ActivityType_dualpa
0	6516	AAA	2014J	4.0	
1	8462	DDD	2013J	0.0	
2	8462	DDD	2014J	0.0	
3	11391	AAA	2013J	0.0	
4	23629	BBB	2013B	0.0	

5 rows × 85 columns

In [38]:
```
studentVle_vle.shape
```

Out[38]: (29228, 85)

In [39]:
```
studentVle_vle[['id_student','code_module','code_presentation']].duplicated()
```

Out[39]: 0

In [40]:
```
student[['id_student','code_module','code_presentation']].duplicated().sum()
```

Out[40]:
```
0
```

In [41]:
```
print(student.isna().sum().sum())
print(studentAssessment_assesment_new.isna().sum().sum())
print(studentVle_vle.isna().sum().sum())
```

```
0
0
0
```

In [42]:
```
studentAssessment_assesment[['id_student','code_module','code_presentation']]
```

Out[42]:
```
148069
```

In [43]:
```
maindf= pd.merge(student, studentVle_vle,  how='left', on=['id_student','code
print("shape of maindf is now: ",maindf.shape)
maindf.head(2)
```

```
shape of maindf is now:  (32593, 97)
```

Out[43]:

	code_module	code_presentation	id_student	gender	region	highest_education	imd_banc
0	AAA	2013J	11391	M	East Anglian Region	HE Qualification	90-100%
1	AAA	2013J	28400	F	Scotland	HE Qualification	20-30%

2 rows × 97 columns

In [44]:
```
# print("Previous shape of maindf is now: ",maindf.shape)
# maindf= pd.merge(maindf, studentAssessment_assesment_new,  how='left', on=[
# print("shape of maindf is now: ",maindf.shape)
# maindf.head(2)

print("Previous shape of maindf is now: ",maindf.shape)
maindf= pd.merge(maindf, studentAssessment_assesment_new,  how='inner', on=['
print("shape of maindf is now: ",maindf.shape)
maindf.head(2)
```

```
Previous shape of maindf is now:  (32593, 97)
shape of maindf is now:  (25843, 151)
```

Out[44]:

	code_module	code_presentation	id_student	gender	region	highest_education	imd_banc
0	AAA	2013J	11391	M	East Anglian Region	HE Qualification	90-100%
1	AAA	2013J	28400	F	Scotland	HE Qualification	20-30%

2 rows × 151 columns

In [52]:
```
pp.ProfileReport(maindf)
```

This variable is highly correlated with `ActivityType_url_Count` *and should be ignored for analysis*

Correlation 0.92941

age_band

Categorical

Distinct count	3
Unique (%)	0.0%
Missing (%)	0.0%
Missing (n)	0

0-35	17947
35-55	7709
55<=	18

In [45]:
```
maindf.shape
```

Out[45]: (25843, 151)

Below Section is just for intuition to know that we have got nearly 6800 rows having Nan value

containing rows when we joined two dataframes to form final one. This was when we did left join, but I have modified the code to inner join so that we do not cheat. Don't understand this ? Check in Report !! It is clearly mentioned why it happend and why I dropped those 6800 rows.

In [46]:
```
# list1=list(maindf.isna().sum(axis=1))
# series = pd.Series(list1)
# result = series.nonzero()
# print(len(result[0]))
# list(result[0])
```

In [47]:
```
# maindf.iloc[list(result[0]),:]['final_result'].value_counts()
```

In [48]:
```
# maindf['final_result'].value_counts()
```

In [49]:
```
# maindf.iloc[list(result[0]),:]
```

Coming back to our original problem : Continuing

In [50]:
```
maindf.isna().sum().sum()
```

Out[50]: 4100

In [51]:
```
maindf.isna().sum()[15:]
```

Out[51]:
```
ActivityType_dataplus_Sum          50
ActivityType_dualpane_Sum          50
ActivityType_externalquiz_Sum      50
ActivityType_folder_Sum            50
ActivityType_forumng_Sum           50
ActivityType_glossary_Sum          50
ActivityType_homepage_Sum          50
ActivityType_htmlactivity_Sum      50
ActivityType_oucollaborate_Sum     50
ActivityType_oucontent_Sum         50
ActivityType_ouelluminate_Sum      50
ActivityType_ouwiki_Sum            50
ActivityType_page_Sum              50
ActivityType_questionnaire_Sum     50
ActivityType_quiz_Sum              50
ActivityType_resource_Sum          50
ActivityType_subpage_Sum           50
ActivityType_url_Sum               50
ActivityType_dataplus_Mean         50
ActivityType_dualpane_Mean         50
ActivityType_externalquiz_Mean     50
ActivityType_folder_Mean           50
ActivityType_forumng_Mean          50
ActivityType_glossary_Mean         50
ActivityType_homepage_Mean         50
ActivityType_htmlactivity_Mean     50
ActivityType_oucollaborate_Mean    50
ActivityType_oucontent_Mean        50
ActivityType_ouelluminate_Mean     50
ActivityType_ouwiki_Mean           50
                                   ..
assessment_type_CMA_Max             0
assessment_type_Exam_Max            0
assessment_type_TMA_Max             0
id_assessment_Min                   0
date_submitted_Min                  0
is_banked_Min                       0
score_Min                           0
date_Min                            0
weight_Min                          0
assessment_type_CMA_Min             0
assessment_type_Exam_Min            0
assessment_type_TMA_Min             0
id_assessment_Count                 0
date_submitted_Count                0
is_banked_Count                     0
score_Count                         0
date_Count                          0
weight_Count                        0
assessment_type_CMA_Count           0
assessment_type_Exam_Count          0
assessment_type_TMA_Count           0
id_assessment_Mode                  0
date_submitted_Mode                 0
is_banked_Mode                      0
```

```
score_Mode                          0
date_Mode                           0
weight_Mode                         0
assessment_type_CMA_Mode            0
assessment_type_Exam_Mode           0
assessment_type_TMA_Mode            0
Length: 136, dtype: int64
```

In [52]:
```
maindf.fillna(0,inplace=True)
```

In [53]:
```
maindf.isna().sum().sum()
```

Out[53]: 0

In [60]:
```
pp.ProfileReport(maindf)
```

Toggle detai

assessment_type_CMA_Sum

Numeric

Distinct count	8
Unique (%)	0.0%
Missing (%)	0.0%
Missing (n)	0
Infinite (%)	0.0%
Infinite (n)	0
Mean	2.7291
Minimum	0
Maximum	7
Zeros (%)	41.6%

In [54]:
```
cols=['ActivityType_dualpane_Count','Count_ALL_Click','ActivityType_ouwiki_Mc
print(len(cols))
maindf.drop(cols,1,inplace=True)
```

54

In [55]:
```
maindf.shape
```

Out[55]: (25843, 100)

In [63]:
```
pp.ProfileReport(maindf)
```

Toggle detai

ActivityType_url_Mode

Boolean

Distinct count	2
Unique (%)	0.0%
Missing (%)	0.0%
Missing (n)	0
Mean	0.00011609

0.0	25840
1.0	

Toggle detai

In [56]:
```
# ['age_band','code_module','code_presentation','disability','gender','highes
# 3+7+4+2+2+5+13=36
# imd_band -- Integer coding by self
#  --- withdraw distinction fail pass
```

In [57]:
```
maindf.date_registration=maindf.date_registration.astype(str).replace('?','0'
# date_unregistration
maindf.date_unregistration=maindf.date_unregistration.astype(str).replace('?'
# week_from_Max
maindf.week_from_Max=maindf.week_from_Max.astype(int)
maindf.week_to_Max=maindf.week_to_Max.astype(int)
maindf.replace({'final_result' : { 'Distinction' : 1,'Pass':1 ,'Withdrawn' :
maindf.imd_band.value_counts()
```

Out[57]:
```
30-40%     2780
20-30%     2749
10-20      2609
40-50%     2553
50-60%     2547
0-10%      2427
60-70%     2388
70-80%     2382
80-90%     2270
90-100%    2140
?           998
Name: imd_band, dtype: int64
```

In [58]:
```
maindf.replace({'imd_band' : { '0-10%' : 1,'10-20':2 ,'20-30%' : 3, '30-40%'
```

In [59]:
```
(maindf.dtypes=='object').sum()
```

Out[59]: 7

In [60]:
```
maindf.shape
```

Out[60]: (25843, 100)

In [61]:
```
# maindf.get_dummies()
opendf=pd.get_dummies(data=maindf, columns=['age_band','code_module','code_pr
opendf.shape
```

Out[61]: (25843, 129)

In [62]:
```
opendf.drop(['id_student'],1,inplace=True)
```

In [63]:
```
opendf.corr()['final_result']
```

Out[63]:

imd_band	0.110237
num_of_prev_attempts	-0.121558
studied_credits	-0.096040
final_result	1.000000
module_presentation_length	0.060302
date_registration	-0.004412
date_unregistration	-0.482324
ActivityType_dataplus_Sum	0.198858
ActivityType_dualpane_Sum	0.150078
ActivityType_externalquiz_Sum	0.100350
ActivityType_folder_Sum	0.160821
ActivityType_forumng_Sum	0.300519
ActivityType_glossary_Sum	0.091901
ActivityType_homepage_Sum	0.504066
ActivityType_htmlactivity_Sum	0.040204
ActivityType_oucollaborate_Sum	0.178433
ActivityType_oucontent_Sum	0.349870
ActivityType_ouelluminate_Sum	0.047413
ActivityType_ouwiki_Sum	0.224509
ActivityType_page_Sum	0.133023
ActivityType_questionnaire_Sum	0.191813
ActivityType_quiz_Sum	0.305255
ActivityType_resource_Sum	0.293516
ActivityType_subpage_Sum	0.294908
ActivityType_url_Sum	0.271929
ActivityType_dataplus_Mean	0.196513
ActivityType_dualpane_Mean	0.033317
ActivityType_externalquiz_Mean	-0.031927
ActivityType_folder_Mean	0.132764
ActivityType_forumng_Mean	0.098597
...	...
code_module_DDD	-0.066684
code_module_EEE	0.077885
code_module_FFF	-0.018045
code_module_GGG	0.073933
code_presentation_2013B	-0.037771
code_presentation_2013J	0.043586
code_presentation_2014B	-0.039273
code_presentation_2014J	0.022080
disability_N	0.063308
disability_Y	-0.063308
gender_F	0.032786
gender_M	-0.032786
highest_education_A Level or Equivalent	0.073681
highest_education_HE Qualification	0.061332
highest_education_Lower Than A Level	-0.121750
highest_education_No Formal quals	-0.028453
highest_education_Post Graduate Qualification	0.029197
region_East Anglian Region	0.014993
region_East Midlands Region	-0.004349
region_Ireland	0.011452
region_London Region	-0.022115
region_North Region	0.021537
region_North Western Region	-0.036388
region_Scotland	-0.000556

```
region_South East Region                          0.026056
region_South Region                               0.032236
region_South West Region                          0.013452
region_Wales                                     -0.028747
region_West Midlands Region                      -0.015863
region_Yorkshire Region                          -0.008665
Name: final_result, Length: 128, dtype: float64
```

In [64]:
```
opendf.to_csv("opendf.csv")
```

Also Our Accuracy Metric would be F1 Score and We can also have a look at Accuracy for corresponding model as it is not highly imbalanced data, so for this classification model, F1 score is "good" to use and accuracy is also "fine" to use.

F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account.

Logistic Regression, KNN, Decision Trees, Random Forest on Full Dataset

80% Training and 20% Testing

Logistic Regression : Final Accuracy : 91.2% and F1 Score of 0.9283

In [375]:
```
X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(np.array(opendf.c
```

In [376]:
```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(max_iter=500,C=0.0008)
logreg.fit(X_train_2, y_train_2)
y_pred_class = logreg.predict(X_test_2)
from sklearn import metrics
print(metrics.accuracy_score(y_test_2, y_pred_class))
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)

0.9121686980073516
```

In [377]:

```
from sklearn.metrics  import f1_score
f1_score(y_test_2, y_pred_class)
```

Out[377]:

```
0.9283685705269801
```

In [378]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
# from sklearn.learning_curve import validation_curve

C_param_range = [0.0006,0.0007,0.0008,0.0009,0.001,0.0011]
trainacclist=[]
testacclist=[]
trainf1score=[]
testf1score=[]
for i in C_param_range:
    # Apply logistic regression model to training data
    lr = LogisticRegression(penalty = 'l2', C = i,random_state = 0)
    lr.fit(X_train_2,y_train_2)
    # Predict using model
    y_pred= lr.predict(X_train_2)
    trainacclist.append(accuracy_score(y_train_2,y_pred))
    trainf1score.append(f1_score(y_train_2, y_pred))
    y_pred= lr.predict(X_test_2)
    testacclist.append(accuracy_score(y_test_2,y_pred))
    testf1score.append(f1_score(y_test_2, y_pred))
```

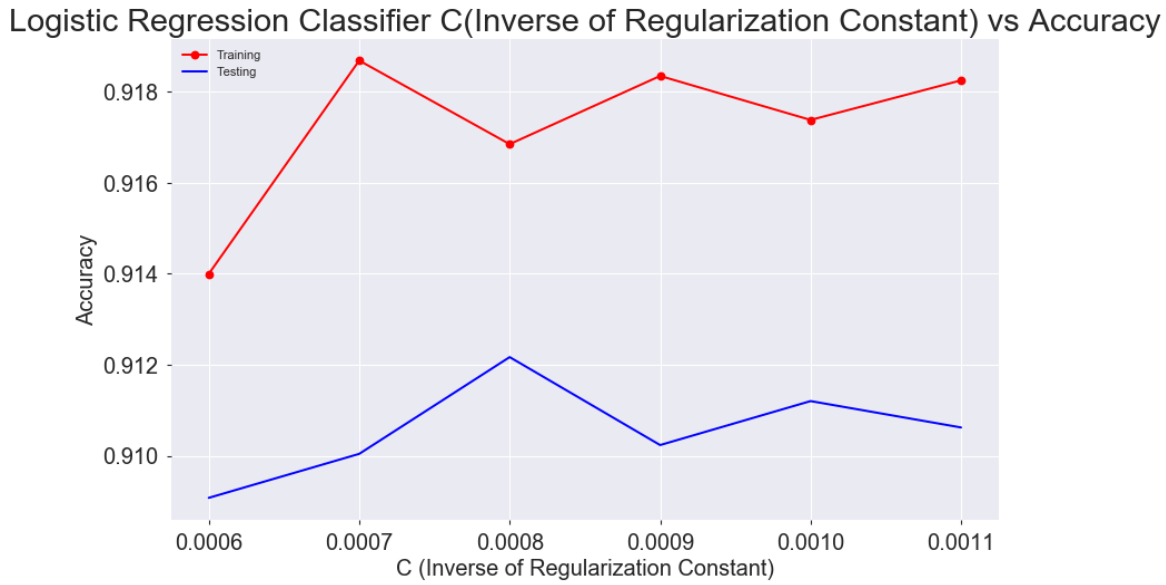
```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
```

In [379]:
```
print(trainacclist)
print(testacclist)
```

```
[0.913998258682403, 0.9186901422076037, 0.916852084744123, 0.91835155267485
74, 0.9173841540098675, 0.9182548128083583]
[0.9090733217256722, 0.910040626813697, 0.9121686980073516, 0.9102340878313
02, 0.9112013929193268, 0.9106210098665118]
```

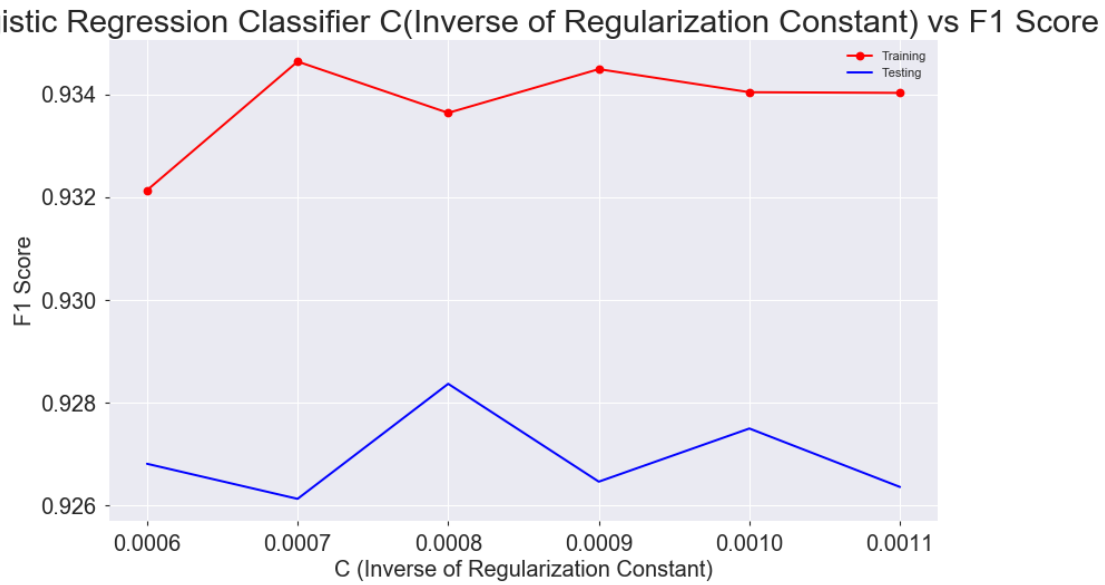
In [380]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(12,7), dpi=80, facecolor='w', edgecolor='k')
plt.xticks(C_param_range,fontsize=18)
plt.yticks(fontsize=18)
plt.title('Logistic Regression Classifier C(Inverse of Regularization Constar
plt.xlabel("C (Inverse of Regularization Constant)",fontsize=18)
plt.ylabel("Accuracy",fontsize=18)
plt.plot(C_param_range,trainacclist,'ro-',label='Training')
plt.plot(C_param_range,testacclist,'b+-',label='Testing')
plt.legend()
```

Out[380]: <matplotlib.legend.Legend at 0x1fb8e4bc828>



In [381]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(12,7), dpi=80, facecolor='w', edgecolor='k')
plt.xticks(C_param_range,fontsize=18)
plt.yticks(fontsize=18)
plt.title('Logistic Regression Classifier C(Inverse of Regularization Constar
plt.xlabel("C (Inverse of Regularization Constant)",fontsize=18)
plt.ylabel("F1 Score",fontsize=18)
plt.plot(C_param_range,trainf1score,'ro-',label='Training')
plt.plot(C_param_range,testf1score,'b+-',label='Testing')
plt.legend()
```

Out[381]: <matplotlib.legend.Legend at 0x1fb8e1f42b0>



K Neighbors Classifier KNN : F1 Score : 0.923 and Accuracy : 90.52%

In [264]:
```
knn = KNeighborsClassifier(n_neighbors=11)
#Fit the model
knn.fit(X_train_2, y_train_2)
#Compute accuracy on the training set
train_accuracy = knn.score(X_train_2, y_train_2)
#Compute accuracy on the test set
test_accuracy = knn.score(X_test_2, y_test_2)
y_pred_class=knn.predict(X_test_2)
```

In [265]:
```
print('Training Accuracy is : ',train_accuracy)
print('Testing Accuracy is : ',test_accuracy)
print('F1 Score is : ',f1_score(y_test_2, y_pred_class))
```

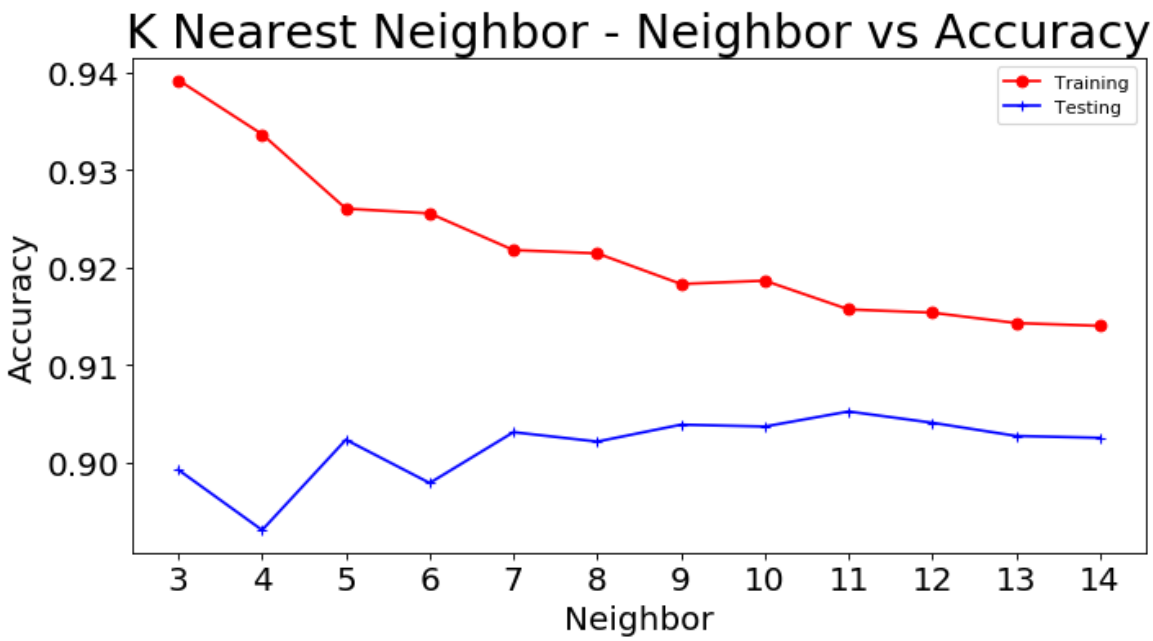
```
Training Accuracy is :  0.9156912063461352
Testing Accuracy is :  0.9052041013735732
F1 Score is :  0.9230043997485857
```

In [261]:

```
from sklearn.neighbors import KNeighborsClassifier

#Setup arrays to store training and test accuracies
neighbors = np.arange(3,15)
train_accuracy =np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))
trainf1=[]
testf1=[]
for i,k in enumerate(neighbors):
    #Setup a knn classifier with k neighbors
    knn = KNeighborsClassifier(n_neighbors=k)

    #Fit the model
    knn.fit(X_train_2, y_train_2)
    y_pred=knn.predict(X_train_2)
    #Compute accuracy on the training set
    train_accuracy[i] = knn.score(X_train_2, y_train_2)
    trainf1.append(f1_score(y_train_2,y_pred))
    y_pred=knn.predict(X_test_2)
    #Compute accuracy on the test set
    test_accuracy[i] = knn.score(X_test_2, y_test_2)
    testf1.append(f1_score(y_test_2,y_pred))
```

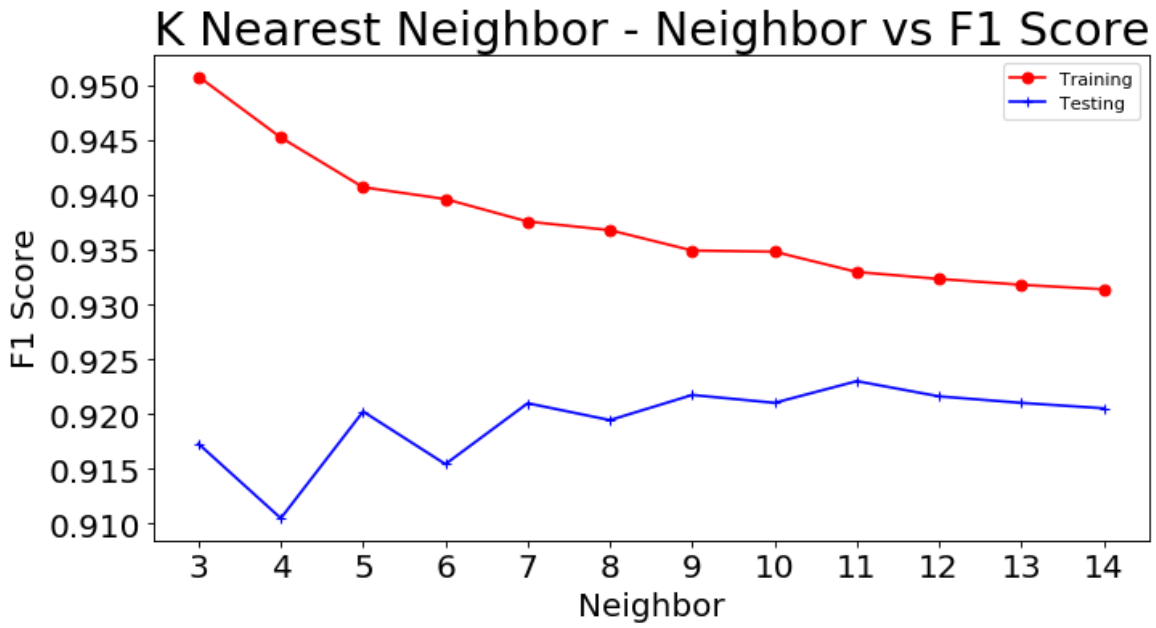
In [262]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')
plt.xticks(np.arange(3,15),fontsize=18)
plt.yticks(fontsize=18)
plt.plot(range(3,15),train_accuracy,'ro-',label='Training')
plt.plot(range(3,15),test_accuracy,'b+-',label='Testing')
plt.title('K Nearest Neighbor - Neighbor vs Accuracy',size=26)
plt.xlabel("Neighbor",fontsize=18)
plt.ylabel("Accuracy",fontsize=18)
plt.legend()
```

Out[262]: <matplotlib.legend.Legend at 0x1fb5b21e470>



In [263]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')
plt.xticks(np.arange(3,15),fontsize=18)
plt.yticks(fontsize=18)
plt.plot(range(3,15),trainf1,'ro-',label='Training')
plt.plot(range(3,15),testf1,'b+-',label='Testing')
plt.title('K Nearest Neighbor - Neighbor vs F1 Score',size=26)
plt.xlabel("Neighbor",fontsize=18)
plt.ylabel("F1 Score",fontsize=18)
plt.legend()
```

Out[263]: <matplotlib.legend.Legend at 0x1fb5b423cf8>



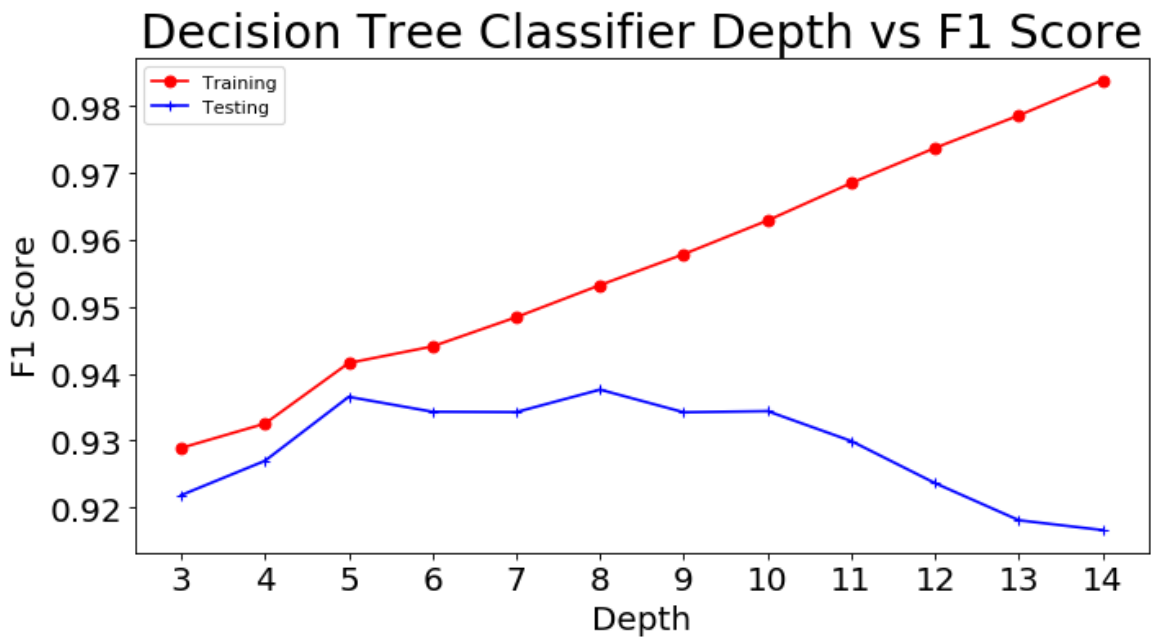
Decision Tree Classifier : F1 Score is : 0.9375
Test Accuracy is : 92.45% at Depth =8

In [266]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 5), dpi=80, facecolor='w', edgecolor='k')

from sklearn.tree import DecisionTreeClassifier
trainf1=[]
testf1=[]
train_accuracy=[]
test_accuracy=[]
for d in range(3,15):
    clf = DecisionTreeClassifier(criterion="gini", max_depth=d)

    # Train Decision Tree Classifer
    clf = clf.fit(X_train_2,y_train_2)
    y_pred=clf.predict(X_train_2)
    trainf1.append(f1_score(y_train_2,y_pred))
    y_pred = clf.predict(X_test_2)
    testf1.append(f1_score(y_test_2,y_pred))
    train_accuracy.append(clf.score(X_train_2, y_train_2))

    #Compute accuracy on the test set
    test_accuracy.append(clf.score(X_test_2, y_test_2) )
```

```
<Figure size 800x400 with 0 Axes>
```

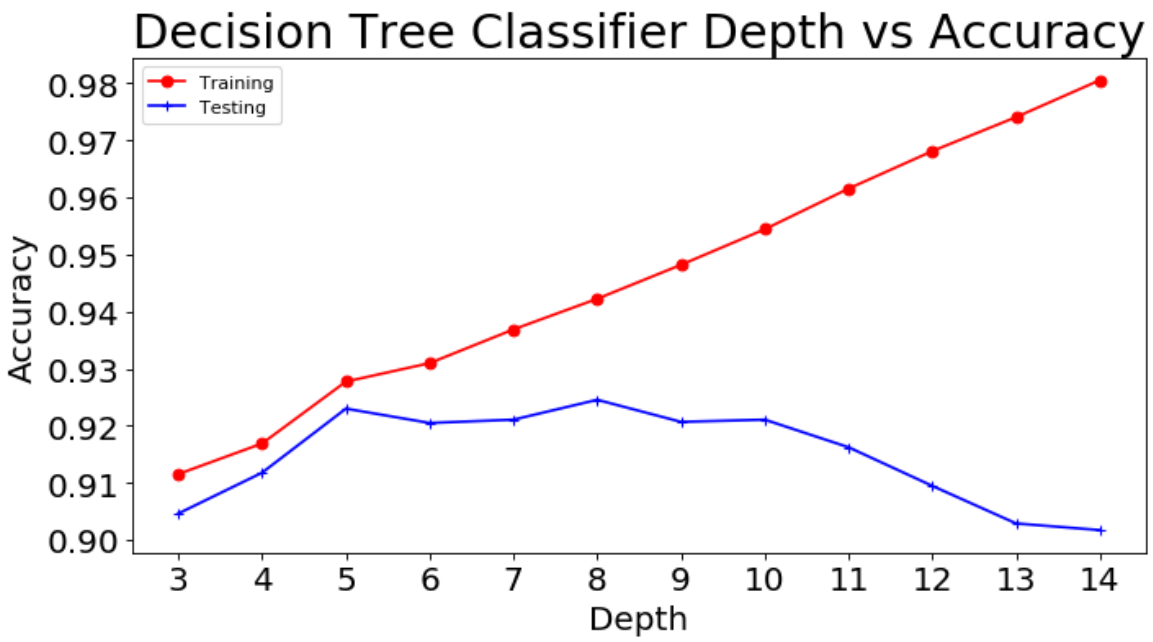
In [269]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')

plt.plot(range(3,15),trainf1,'ro-',label='Training')
plt.xticks(np.arange(3,15),fontsize=18)
plt.yticks(fontsize=18)
plt.title('Decision Tree Classifier Depth vs F1 Score',size=26)
plt.plot(range(3,15),testf1,'b+-',label='Testing')
plt.xlabel("Depth",fontsize=18)
plt.ylabel("F1 Score",fontsize=18)
plt.legend()
```

Out[269]: <matplotlib.legend.Legend at 0x1fb7e9f5c50>



In [273]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')

plt.plot(range(3,15),train_accuracy,'ro-',label='Training')
plt.xticks(np.arange(3,15),fontsize=18)
plt.yticks(fontsize=18)
plt.title('Decision Tree Classifier Depth vs Accuracy',size=26)
plt.plot(range(3,15),test_accuracy,'b+-',label='Testing')
plt.xlabel("Depth",fontsize=18)
plt.ylabel("Accuracy",fontsize=18)
plt.legend()
```

Out[273]: <matplotlib.legend.Legend at 0x1fb7fda1da0>



In [276]:
```
print("At Depth = 8 ")
print("Test Accuracy is :",test_accuracy[5])
print("F1 Score is :",testf1[5])
```

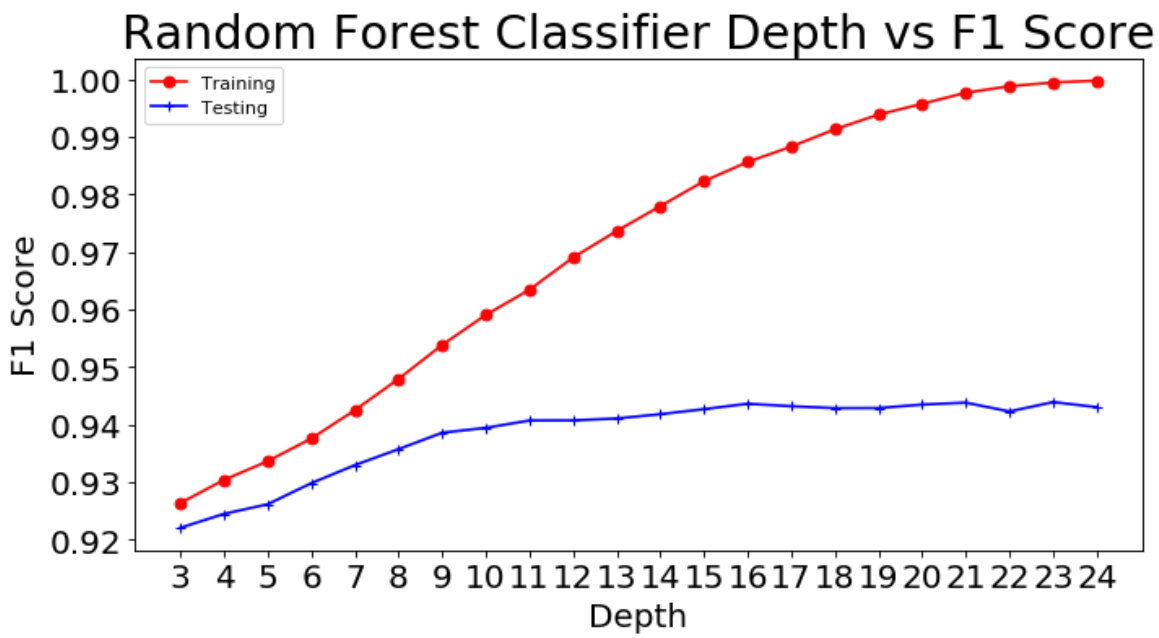
```
At Depth = 8
Test Accuracy is : 0.9245502031340684
F1 Score is : 0.9375999999999999
```

Random Forest Classifier: At Depth = 16, f1 score of 0.9436 and Testing Accuracy of 93.13

In [281]:
```
trainf1=[]
testf1=[]
train_accuracy=[]
test_accuracy=[]
i=0
for d in range(3,25):
    clf = RandomForestClassifier(n_estimators=200, random_state=42, max_depth
    # Train Decision Tree Classifer
    clf = clf.fit(X_train_2,y_train_2)
    y_pred=clf.predict(X_train_2)
    trainf1.append(f1_score(y_train_2,y_pred))
    y_pred = clf.predict(X_test_2)
    testf1.append(f1_score(y_test_2,y_pred))
    train_accuracy.append(clf.score(X_train_2, y_train_2))

    #Compute accuracy on the test set
    test_accuracy.append(clf.score(X_test_2, y_test_2) )
    i+=1
```

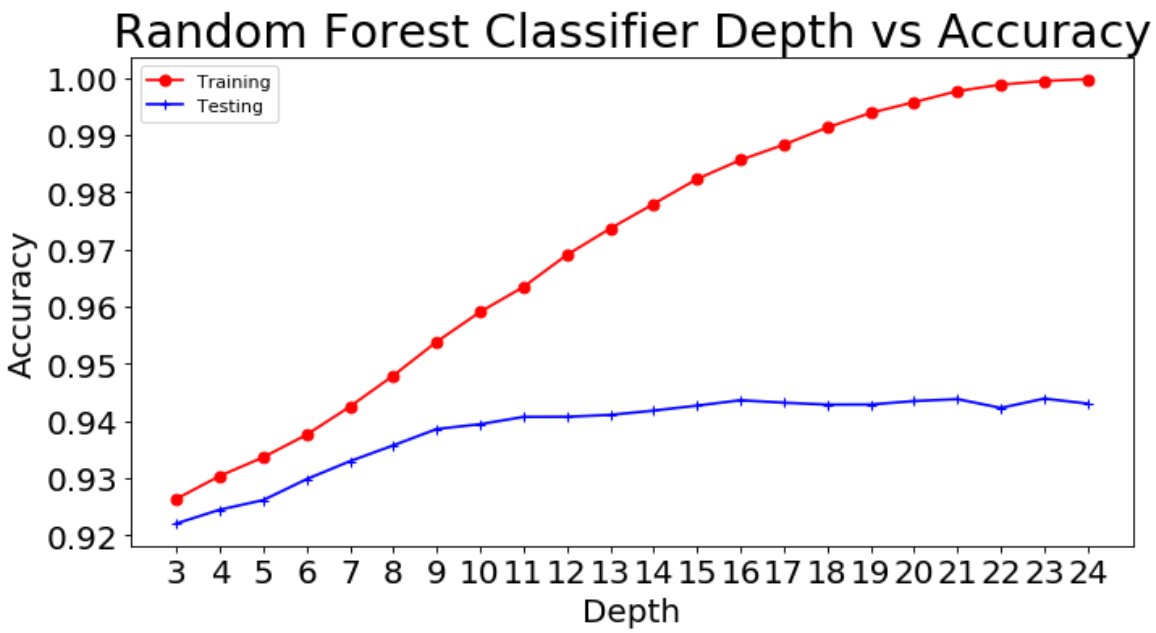
In [294]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')

plt.plot(range(3,25),trainf1,'ro-',label='Training')
plt.xticks(np.arange(3,25),fontsize=18)
plt.yticks(fontsize=18)
plt.title('Random Forest Classifier Depth vs F1 Score',size=26)
plt.xlabel("Depth",fontsize=18)
plt.ylabel("F1 Score",fontsize=18)
plt.plot(range(3,25),testf1,'b+-',label='Testing')
plt.legend()
```

Out[294]: <matplotlib.legend.Legend at 0x1fb808cd240>



In [295]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10,5), dpi=80, facecolor='w', edgecolor='k')

plt.plot(range(3,25),trainf1,'ro-',label='Training')
plt.xticks(np.arange(3,25),fontsize=18)
plt.yticks(fontsize=18)
plt.title('Random Forest Classifier Depth vs Accuracy',size=26)
plt.xlabel("Depth",fontsize=18)
plt.ylabel("Accuracy",fontsize=18)
plt.plot(range(3,25),testf1,'b+-',label='Testing')
plt.legend()
```

Out[295]:
```
<matplotlib.legend.Legend at 0x1fb808f0710>
```



In [296]:
```
print(" At Depth = 16, Testing Accuracy of ",test_accuracy[13], "and f1 score
```

```
 At Depth = 16, Testing Accuracy of  0.9313213387502418 and f1 score of  0.
943623947911704
```

In [402]:
```
## Import the random forest model.
from sklearn.ensemble import RandomForestClassifier
## This line instantiates the model.
rf = RandomForestClassifier(n_estimators=200, random_state=42, max_depth=16)
## Fit the model on your training data.
rf.fit(X_train_2, y_train_2)
## And score it on your testing data.
rf.score(X_test_2, y_test_2)
```

Out[402]:
```
0.9313213387502418
```

In [403]:
```
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
ypred=rf.predict(X_test_2)
print(classification_report(y_test_2, ypred))
print(confusion_matrix(y_test_2, ypred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.85      0.91      2157
           1       0.90      0.99      0.94      3012

   micro avg       0.93      0.93      0.93      5169
   macro avg       0.94      0.92      0.93      5169
weighted avg       0.94      0.93      0.93      5169

[[1843  314]
 [  41 2971]]
```

Relative Feature Importances of Features: Best 15 Features for Model

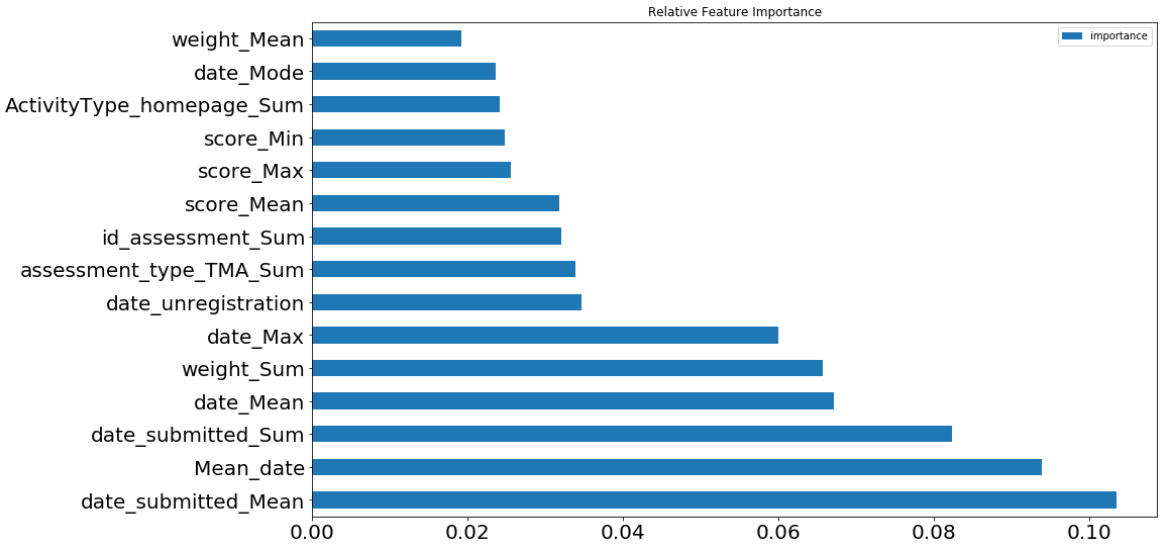
In [301]:
```
import pandas as pd
feature_importances = pd.DataFrame(rf.feature_importances_,index = opendf.drc
feature_importances[:15]
```

Out[301]:

	importance
date_submitted_Mean	0.103519
Mean_date	0.093889
date_submitted_Sum	0.082346
date_Mean	0.067122
weight_Sum	0.065707
date_Max	0.059974
date_unregistration	0.034619
assessment_type_TMA_Sum	0.033904
id_assessment_Sum	0.032058
score_Mean	0.031788
score_Max	0.025609
score_Min	0.024803
ActivityType_homepage_Sum	0.024174
date_Mode	0.023629
weight_Mean	0.019231

In [329]:
```
rtances[:15].plot(kind='barh', stacked=True,figsize=(15,9),title="Relative Fe
```

Out[329]: <matplotlib.axes._subplots.AxesSubplot at 0x1fb8c773da0>



XGBOOST MODEL:

In [353]:

```
#Creating Train Test split from here. X_test and y_test is TEST SET and they
X_train, X_test, y_train, y_test = train_test_split(np.array(opendf.drop('fir

# X_train and y_train will be further splitted into X_train_new and X_valid c
X_train_new, X_validation, y_train_new, y_validation = train_test_split(X_tra

# Creating Evaluation Set using X_validation and y_validation which we create
validation_set = [ ( X_validation, y_validation ) ]
print('Training Features Shape:', X_train.shape)
print('Training Label Shape:', y_train.shape)
print('Testing Features Shape:', X_test.shape)
print('Testing Label Shape:', y_test.shape)
f1=[]
acc=[]
for l in [0.075,0.076,0.077,0.078,0.079,0.08,0.081,0.082,0.083,0.084]:
    for d in [10,11,12,13,14,15,16,17,18,19,20,21,22]:
        xgbmodel = xgb.XGBClassifier(base_score=0.5, booster='gbtree', colsam
               colsample_bytree=0.5, gamma=0, learning_rate=l, max_delta_step
               max_depth=d, min_child_weight=3, missing=None, n_estimators=10
               n_jobs=1, nthread=4, objective='binary:logistic', random_state
               reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=1337, sile
               subsample=0.9)

        xgbmodel.fit( X_train_new, y_train_new, eval_metric="auc",early_stopp
        ypred=xgbmodel.predict(X_test)
        score = accuracy_score(y_test,ypred)
        f1score=f1_score(y_test,ypred)
        f1.append(f1score)
        acc.append(score)
        print("For learning rate :  = ",l)
        print("Accuracy is : ",score)
        print("F1 Score is : ",f1score,"\n\n\n")
```

```
[73]    validation_0-auc:0.979264
[74]    validation_0-auc:0.979248
[75]    validation_0-auc:0.979249
[76]    validation_0-auc:0.979266
[77]    validation_0-auc:0.979292
[78]    validation_0-auc:0.979196
[79]    validation_0-auc:0.979073
[80]    validation_0-auc:0.979096
[81]    validation_0-auc:0.979048
[82]    validation_0-auc:0.979096
[83]    validation_0-auc:0.979154
[84]    validation_0-auc:0.979177
[85]    validation_0-auc:0.979151
[86]    validation_0-auc:0.979138
[87]    validation_0-auc:0.97908
[88]    validation_0-auc:0.979131
[89]    validation_0-auc:0.979144
[90]    validation_0-auc:0.979112
[91]    validation_0-auc:0.97913
[92]    validation_0-auc:0.979139
```

In [354]:
```
max(acc)
```

Out[354]: 0.9315280464216634

In [355]:
```
max(f1)
```

Out[355]: 0.9421000981354268

XGBOOST MODEL BEST :F1 Score = 0.9421 and Accuracy = 0.9315

Parameters: Best Learning rate 0.081 , Depth = 10

In [372]:

```
#Creating Train Test split from here. X_test and y_test is TEST SET and they
X_train, X_test, y_train, y_test = train_test_split(np.array(opendf.drop('fir

# X_train and y_train will be further splitted into X_train_new and X_valid c
X_train_new, X_validation, y_train_new, y_validation = train_test_split(X_tra

# Creating Evaluation Set using X_validation and y_validation which we create
validation_set = [ ( X_validation, y_validation ) ]
print('Training Features Shape:', X_train.shape)
print('Training Label Shape:', y_train.shape)
print('Testing Features Shape:', X_test.shape)
print('Testing Label Shape:', y_test.shape)
xgbmodel = xgb.XGBClassifier(base_score=0.5, booster='gbtree', colsample_byle
           colsample_bytree=0.5, gamma=0, learning_rate=0.081, max_delta_step
           max_depth=10, min_child_weight=3, missing=None, n_estimators=100,
           n_jobs=1, nthread=4, objective='binary:logistic', random_state=0,
           reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=1337, silent=1
           subsample=0.9)
xgbmodel.fit( X_train_new, y_train_new, eval_metric="auc",early_stopping_roun
ypred=xgbmodel.predict(X_test)
score = accuracy_score(y_test,ypred)
f1score=f1_score(y_test,ypred)
f1.append(f1score)
acc.append(score)
print("For learning rate :  = ",0.081)
print("Accuracy is : ",score)
print("F1 Score is : ",f1score,"\n\n\n")
```

```
[65]    validation_0-auc:0.979419
[66]    validation_0-auc:0.97942
[67]    validation_0-auc:0.979493
[68]    validation_0-auc:0.979496
[69]    validation_0-auc:0.979495
[70]    validation_0-auc:0.979546
[71]    validation_0-auc:0.979561
[72]    validation_0-auc:0.979534
[73]    validation_0-auc:0.979614
[74]    validation_0-auc:0.979628
[75]    validation_0-auc:0.979639
[76]    validation_0-auc:0.979631
[77]    validation_0-auc:0.979627
[78]    validation_0-auc:0.979588

[79]    validation_0-auc:0.97956
[80]    validation_0-auc:0.979568
[81]    validation_0-auc:0.979595
[82]    validation_0-auc:0.97966
[83]    validation_0-auc:0.979647
```

In [373]:
```
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
print(classification_report(y_test, ypred))
print(confusion_matrix(y_test, ypred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.88      0.92      1104
           1       0.91      0.97      0.94      1481

   micro avg       0.93      0.93      0.93      2585
   macro avg       0.94      0.92      0.93      2585
weighted avg       0.93      0.93      0.93      2585

[[ 968  136]
 [  41 1440]]
```

In [374]:
```
xgbmodel.feature_importances_
d={}
for i in range(0,opendf.drop('final_result',1).shape[1]-1):
    d[opendf.drop('final_result',1).columns[i]]=xgbmodel.feature_importances_
d=sorted(d.items(),key= lambda value: value[1],reverse=True)
d
```

Out[374]:
```
[('date_submitted_Mean', 0.18771051),
 ('date_unregistration', 0.11053139),
 ('date_Mean', 0.036873545),
 ('weight_Sum', 0.035936706),
 ('assessment_type_CMA_Sum', 0.033528946),
 ('Mean_date', 0.031892624),
 ('date_submitted_Sum', 0.024254277),
 ('assessment_type_Exam_Sum', 0.022789065),
 ('score_Mean', 0.020924052),
 ('date_Mode', 0.01903229),
 ('assessment_type_TMA_Sum', 0.018014407),
 ('weight_Min', 0.017776038),
 ('code_module_FFF', 0.016504662),
 ('score_Min', 0.01456422),
 ('code_module_CCC', 0.0141060725),
 ('weight_Mode', 0.0130035775),
 ('id_assessment_Sum', 0.012311276),
 ('code_module_DDD', 0.011018931),
 ('weight_Mean', 0.010848449),
```

In [400]:
```
p=pd.DataFrame(d)
p.index=pd.DataFrame(d)[0]
p.drop(0,1,inplace=True)
p.columns=['importances']
p[:15]
```

Out[400]:

0	importances
date_submitted_Mean	0.187711
date_unregistration	0.110531
date_Mean	0.036874
weight_Sum	0.035937
assessment_type_CMA_Sum	0.033529
Mean_date	0.031893
date_submitted_Sum	0.024254
assessment_type_Exam_Sum	0.022789
score_Mean	0.020924
date_Mode	0.019032
assessment_type_TMA_Sum	0.018014
weight_Min	0.017776
code_module_FFF	0.016505
score_Min	0.014564
code_module_CCC	0.014106

In [401]:
```
p[:15].plot(kind='barh', stacked=True,figsize=(15,9),title="Relative Feature
```

Out[401]: <matplotlib.axes._subplots.AxesSubplot at 0x1fb9edf7668>