Econ691-06 | Hanh Nong

ASSIGNMENT 1

In [1]:
```
#setup
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.formula.api as sfa
from statsmodels.iolib.summary2 import summary_col
from scipy import stats
```

Problem 1: Treatment Effects

1. (10 points)

Compute each of the ATE, ATT, and ATU.

In [4]:
```
# data from table
Y1 = np.array([22, 26, 20, 23, 21, 17, 35, 33, 27, 42])
Y0 = np.array([12, 14, 18, 16, 19, 18, 30, 31, 28, 36])
d = np.array([1, 1, 1, 1, 1, 1, 0, 0, 0, 0,])
Y = Y1 - Y0
```

$$ATE = E(Y^1 - Y^0) = 4.4$$

In [9]:
```
# working for ATE
ate = Y.mean()
ate
```

Out[9]: 4.4

$$ATT = E(Y^1|\delta = 1) - E(Y^0|\delta = 1) = 5.3$$

In [22]:
```
# working for ATT
att = Y1[np.where(d == 1)].mean() - Y0[np.where(d == 1)].mean()
round(att,2)
```

Out[22]: 5.33

$$ATU = E(Y^1|\delta = 0) - E(Y^0|\delta = 0) = 3$$

In [11]:
```
# working for ATU
atu = Y1[np.where(d == 0)].mean() - Y0[np.where(d == 0)].mean()
atu
```

Out[11]: 3.0

In [24]:
```
#checking
p = sum(d)/len(d) #proportion of treatment group
round((att*p + atu*(1-p)) - ate)
```

Out[24]: -0.0

2. (5 points)

Showing all work, derive a single equation that links these three concepts to each other, observed data, and selection bias.

Observed data = ATE + Selection bias + Heterogeneity bias

$\rightarrow \overline{Y}$ of treatment group - $\overline{Y}$ of control group = [p * ATT + (1-p) * ATU] + [$\overline{Y}$ of treatment group if not treated - $\overline{Y}$ of control group] + [(1-p) * (ATT - ATU)]

$$\rightarrow E(Y^1|\delta = 1) - E(Y^0|\delta = 0) = E(Y^1 - Y^0) + [E(Y^0|\delta = 1) - E(Y^0|\delta = 0)] + (1 - p) * [E(Y^1|\delta = 0) - E(Y^0|\delta = 0)]$$

$$\rightarrow -9.75 \approx 4.4 - 15.08 + 1.2$$

Note: p is the proportion of treatment group

In [39]:
```
nate = Y1[np.where(d==1)].mean() - Y0[np.where(d==0)].mean()
selBias = Y0[np.where(d==1)].mean() - Y0[np.where(d==0)].mean()
hetBias = (1-p)*(Y1[np.where(d==0)].mean()-Y0[np.where(d==0)].mean())

print("Naive ATE:",nate)
print("Selection bias:",round(selBias,2))
print("Heterogeneity bias:",round(hetBias,2))

#checking
round(nate - (ate + selBias + hetBias),2)
```

```
Naive ATE: -9.75
Selection bias: -15.08
Heterogeneity bias: 1.2
```

Out[39]:
```
-0.27
```

3. (10 points)

In practice, which part of this equation is observable; and what value do you get? Which part of this equation is likely of interest; and what value do you get?

The left hand side of this equation is observable. It is the naive calculation of ATE by taking the difference between outcomes of applying treatment on treatment group and outcomes of not applying treatment on control group. The value for this part is -9.75 (refer to variable *nate* in code chunk of question 2)

The part should be of interest is ATE on the right hand side of the equation. This is the true treatment effect. The value of ATE is 4.4 (see variable *ate* in code chunk of question 1)

4. (10 points)

Assuming that your empirical estimates are not due to sampling noise, which assumptions in the classic experimentation model are likely violated in this table? Does this explain the discrepancy in Question 3, and how does each assumption failure increase or decrease the discrepancy?

The discrepance in question 3 is -14.15. This discrepancy can be explained by the possible failure of the following assumptions in the classic experimentation model:

- ***Constant treatment effect:*** this assumes treatment effect is the same across sample, in other word $\Delta Y_i = Y_i^1 - Y_i^0 =$ constant for all units in the experiment. When this assumption is not met, which is clearly in this case (see values of variable *Y* in code below), ATT $\neq$ ATU which led to non-zero value of heterogeneity bias. Below is how this failure increase the discrepancy:
  - $ATT = E(Y^1|\delta = 1) - E(Y^0|\delta = 1) = E(Y^1 - Y^0|\delta = 1) = E(\Delta Y|\delta = 1) = 5.3$
  - $ATU = E(Y^1|\delta = 0) - E(Y^0|\delta = 0) = E(Y^1 - Y^0|\delta = 0) = E(\Delta Y|\delta = 0) = 3$
  - $\Delta Y \neq \text{constant} \rightarrow E(\Delta Y|\delta = 1) \neq E(\Delta Y|\delta = 0) \rightarrow ATT \neq ATU$
  - Heterogeneity bias = (1-p) * (ATT - ATU)$\rightarrow$ Heterogeneity bias $= 1.2 \neq 0$
  - Nevertheless, it doesn't mean that failure in this assumption is fully accountable for the value of heterogeneity bias.

- ***Independent assignment of treatment:*** ideally, treatment should be assigned to units independently from the potential outcomes. Taking the stand point that potential outcomes should be fixed and assignment of treatment should not be fixed but rather random. If treatment is not assigned independently from the potential outcomes, then the potential outcomes of the treatment group will differ from potential outcomes of the control group had this treatment applied. Similarly, potential outcomes of treatment group had this treatment not applied will differ from the potential outcomes of the control group. When this assumption fails, we have:
  - (1) $E(Y^1|\delta = 1) \neq E(Y^1|\delta = 0) \rightarrow E(Y^1|\delta = 1) - E(Y^1|\delta = 0) \neq 0$ (these are first terms in ATT and ATU respectively), and
  - (2) $E(Y^0|\delta = 1) \neq E(Y^0|\delta = 0) \rightarrow E(Y^0|\delta = 1) - E(Y^0|\delta = 0) \neq 0$ (these are second terms in ATT and ATU respectively)

  Failure in this assumption will lead to both selection bias and heterogeneity bias:
  - Selection bias: the left hand side of inequality (2) is the selection bias (see question 2). The value of selection bias is -15.08, a decrease in the discrepancy mentioned in question 3.
  - Heterogeneity bias: (1-p) * (ATT - ATU).

    ATT - ATU = $[E(Y^1|\delta = 1) - E(Y^0|\delta = 1)] - [E(Y^1|\delta = 0) - E(Y^0|\delta = 0)]$

    $= E(Y^1|\delta = 1) - E(Y^1|\delta = 0) - [E(Y^0|\delta = 1) - E(Y^0|\delta = 0)]$

    = LHS of inequality (1) + LHS of inequality (2) $\neq 0$

- SUTVA: the assumption that assignment of the treatment to one unit cannot affect the treatment effect or the potential outcomes of another unit. If this assumption fails, the heterogeneity bias is likely to be impact.

In [37]:
```
Y
```

Out[37]:
```
array([10, 12,  2,  7,  2, -1,  5,  2, -1,  6])
```

Problem 2: Analysis on Classic Experiments

Suppose you are a data scientist at TikTok, and you are considering deploying a feature - richer color palettes - to increase time spent on the app. You run a small and short experiment to test its impact, with an equal treatment and control group.

The data following the experiment is in the file data assignment1.csv. The file has four columns and 1200 rows. Each row corresponds to a separate user, for which there are four columns: time spent on Tiktok (in seconds) during the experiment, an indicator for whether the user is in the treatment group (where 1 indicates treatment), the age of the user, and the number of channels (i.e. other accounts) to which the user previously subscribed.

In [2]:
```
#Load data
data = pd.read_csv("data_assignment1.csv")
data.head()
```

Out[2]:

	timespent	treatment	age	channels
0	228.565845	1	20	3
1	167.198848	1	27	0
2	142.921240	1	29	3
3	305.336382	1	13	6
4	170.465854	1	25	5

1. (10 points)

Without using regression techniques, what is your assessment of the experiment including both its estimated impact and the significance of that estimate? To assess significance, use both analytic techniques and the bootstrap. Be very clear about precisely how you performed the bootstrap, and why you chose to do it that way.

Among different ages and number of channels, there is great fluctuation in effect as shown in the above graph. The control/treatment split is around 50% and this split is also consistent among each category of age and number of channels. In all age group, number of users who are 12 is much larger than all other age group. This probably is due to this age group is the main user segment of TikTok.

Using t-test:

- Null hypothesis: Mean of timespent of treatment group - Mean of timepspent of control group = 0
- Alternative hypothesis: Mean of timespent of treatment group - Mean of timepspent of control group > 0
- Significant level set at 5%

T-test returns the T score of 1.38 and p-value of 0.167 which is much higher than 5%. The estimated impact of the experiment is not statistically significant.

Overall, the estimated impact is an increase in time spent of 4.24 seconds when using the new color pallettes (see variable *te* in code below). This sounds like there is an effect of treatment.

In [58]:
```
#Calculate treatment effect:
te = data.query('treatment == 1')['timespent'].mean() - data.query('treatment == 0')['timespent'].mean()
print("Overall effect of the experiment:",round(te,2))
```

```
Overall effect of the experiment: 4.24
```

T-test

T-test is run in order to assess the significance of the result above. This t-test test whether there is significance difference between the means of two samples. With the significant set at 5%, the high value of p-value (0.1667) indicates this result is not statistically significant.

In [95]:
```
t, p = stats.ttest_ind(data.query('treatment ==1')['timespent'],data.query('treatment ==0')['timespent'])
print("t score:", round(t,3))
print("p-value:", round(p,3))
```

```
t score: 1.383
p-value: 0.167
```

Bootstrap

For question 1, a simple bootstrap approach is used to assess the significance of the result. This approach assumes total randomness in the number of a particular age and channel value in each control and treatment group. This bootstrap preserve the exact number of units in each control and treatment groups.

In the original experiment, there are 600 control units and 600 treatment units. This could be understood in two ways:

- there are exactly 600 control units and 600 treatment units; or
- units are assigned into each group with the chance of 50% and there is no difference in total number of units in each group due to randomness.

For a population of 1,200 units, it is hard to ensure that there would be no random difference in number of units in each group. An intentional 600 control units and 600 treatment units is more believable. Moreover, by retaining this exact allocation, it would align with the t-test above for comparison purpose. Hence, this bootstrap goes with this exact allocation of control and treatment.

The following code is to run bootstrap:

In [46]:
```
mean_diff1 = []
reg_bstr_1 = []
regSimp_bstr_1 = []
regCmplx_bstr_1= []
n1_control = int(data.query('treatment == 0').shape[0])
n1_treatment = int(data.query('treatment == 1').shape[0])

for i in range(1000):

#BOOTSTRAP APPROACH 1: Number of units in each treatment and control groups is preserved.
    index_control = np.random.choice(np.where(data['treatment']==1)[0], replace = True, size = n1_control)
    index_treatment = np.random.choice(np.where(data['treatment']==0)[0], replace = True, size = n1_treatment
)
    sample1 = pd.concat([data.iloc[index_control], data.iloc[index_treatment]], ignore_index=True, sort=False
)

    sample1['agesq'] = sample1['age']**2
    sample1['chansq'] = sample1['channels']**2
    sample1['dxchan'] = sample1['treatment']*sample1['channels']

    #for Q.1
    mean_diff1.append(sample1.query('treatment == 1')['timespent'].mean() -
                      sample1.query('treatment == 0')['timespent'].mean())
    #for Q.2
    reg_bstr_1.append(sfa.ols("timespent ~ treatment", data=sample1)
                          .fit(cov_type='HC1')
                          .params[1])
    #for Q.3
    regSimp_bstr_1.append(sfa.ols("timespent ~ treatment + age + channels", data=sample1)
                          .fit(cov_type='HC1')
                          .params[1])
    #for Q.4
    regCmplx_bstr_1.append(sfa.ols("timespent ~ treatment + age + channels + agesq + chansq + dxchan",
                                   data=sample1)
                           .fit(cov_type='HC1')
                           .params[1])
```

Result of bootstrap show an effect of 4.3 which aligns with the experiment. The standard error of 3 is very high for this value which indicates this effect is not significant. This conclusion is also consistent with t-test.

In [47]:
```
print("Mean of effect:",round(np.mean(mean_diff1),3))
print("std of mean difference:",round(np.std(mean_diff1), 3))
```

```
Mean of effect: 4.3
std of mean difference: 3.062
```

2. (10 points)

Using regression techniques but without using the additional covariates, what is your assessment? Again, use both analytic techniques and the bootstrap to assess significance; and be clear about the bootstrap approach.

Regression: simple model

In [49]:
```
reg = sfa.ols("timespent ~ treatment", data=data).fit(cov_type='HC1')
print(reg.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              timespent   R-squared:                       0.002
Model:                            OLS   Adj. R-squared:                  0.001
Method:                 Least Squares   F-statistic:                     1.914
Date:                Mon, 31 Aug 2020   Prob (F-statistic):              0.167
Time:                        17:58:01   Log-Likelihood:                -6469.1
No. Observations:                1200   AIC:                         1.294e+04
Df Residuals:                    1198   BIC:                         1.295e+04
Df Model:                           1
Covariance Type:                  HC1
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    225.6296      2.126    106.118      0.000     221.462     229.797
treatment      4.2438      3.068      1.383      0.167      -1.769      10.256
==============================================================================
Omnibus:                      358.879   Durbin-Watson:                   1.981
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               56.565
Skew:                          -0.044   Prob(JB):                     5.21e-13
Kurtosis:                       1.940   Cond. No.                         2.62
==============================================================================

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC1)
```

The results in regression is quite similiar with what calculated in question 1: effect of 4.24 with standard error of 3 and p-value of 0.167. The R^2 is less than 1%. These results indicate the effect is not significant. This model is too simple to exlain the effect.

In [50]:
```
print("Coefficient of treatment:",round(reg.params[1],2))
print("Standard error of coefficient:",round(reg.bse[1],2))
```

```
Coefficient of treatment: 4.24
Standard error of coefficient: 3.07
```

Bootstrap

The same approach of bootstrap is used as in question 1. Regression with the same formula for the simple model is run on bootstrap sample for each iteration. With this approach, there should be no difference in the results compared with question 1: Mean of effect is 4.3 and standard error of 3. With the standard error of 70% of the effect itself, this effect can't be significant

In [51]:
```
print("Mean of effect, simple model, bootstrap approach 1:",round(np.mean(reg_bstr_1),3))
print("std of simple model's coefficient, bootstrap approach 1:",round(np.std(reg_bstr_1), 3))
```

```
Mean of effect, simple model, bootstrap approach 1: 4.3
std of simple model's coefficient, bootstrap approach 1: 3.062
```

3. (5 points)

Now incorporate the additional covariates in your regression model. What is your assessment of the experiment?

Additional variables added are age and channels

In [35]:
```
regSimp = sfa.ols("timespent ~ treatment + age + channels", data=data).fit(cov_type='HC1')
print(regSimp.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              timespent   R-squared:                       0.942
Model:                            OLS   Adj. R-squared:                  0.942
Method:                 Least Squares   F-statistic:                     6156.
Date:                Mon, 31 Aug 2020   Prob (F-statistic):               0.00
Time:                        15:27:01   Log-Likelihood:                -4760.5
No. Observations:                1200   AIC:                             9529.
Df Residuals:                    1196   BIC:                             9549.
Df Model:                           3
Covariance Type:                  HC1
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    412.4169      1.649    250.092      0.000     409.185     415.649
treatment      3.0486      0.739      4.123      0.000       1.599       4.498
age           -9.2744      0.069   -135.061      0.000      -9.409      -9.140
channels       0.9735      0.114      8.566      0.000       0.751       1.196
==============================================================================
Omnibus:                        2.097   Durbin-Watson:                   2.053
Prob(Omnibus):                  0.350   Jarque-Bera (JB):                2.143
Skew:                           0.080   Prob(JB):                        0.343
Kurtosis:                       2.869   Cond. No.                         93.4
==============================================================================

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC1)
```

The effect of treatment has decreased to 3. P-value is very low and the standard error has decreased. R^2 increased 0.94. Including age and channels to the regression has significant impact to the model. Older ages lead to adverse impact to the effect. Channels also has a role in the time spent on the app. In this model, the effect of 3 is a significant result. This model does a better job in explaining the effect compared to the model in question 2. The result of 4.24 calculated in question 1 includes bias which was not shown in the initial regression.

4. (10 points)

Another data scientist at TikTok tells you that your model in Question 3 is too simple. She tells you that, rather than using the covariates as simple linear predictors, you should include some non-linear transformations of those covariates. Specifically, she recommends that you use the square of channels, the square of age, and the interaction between channels and treatment as additional covariates. You're less sure, as this model is substantially more complex and may have higher variance. Using the bootstrap only, compare the variance of the complex model versus your simple model.

Regression: Complex model

In [38]:
```
data['agesq'] = data['age']**2
data['chansq'] = data['channels']**2
data['dxchan'] = data['treatment']*data['channels']
```

In [60]:
```
regCmplx = sfa.ols("timespent ~ treatment + age + channels + agesq + chansq + dxchan", data=data).fit(cov_type='HC1')
print(regCmplx.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              timespent   R-squared:                       0.943
Model:                            OLS   Adj. R-squared:                  0.943
Method:                 Least Squares   F-statistic:                     3157.
Date:                Mon, 31 Aug 2020   Prob (F-statistic):               0.00
Time:                        18:12:54   Log-Likelihood:                -4750.1
No. Observations:                1200   AIC:                             9514.
Df Residuals:                    1193   BIC:                             9550.
Df Model:                           6
Covariance Type:                  HC1
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    439.1900      6.313     69.574      0.000     426.818     451.562
treatment      2.7396      1.364      2.008      0.045       0.065       5.414
age          -11.9522      0.613    -19.488      0.000     -13.154     -10.750
channels       0.6384      0.445      1.434      0.152      -0.234       1.511
agesq          0.0638      0.014      4.430      0.000       0.036       0.092
chansq         0.0297      0.042      0.713      0.476      -0.052       0.111
dxchan         0.0450      0.225      0.200      0.842      -0.396       0.486
==============================================================================
Omnibus:                        1.277   Durbin-Watson:                   2.059
Prob(Omnibus):                  0.528   Jarque-Bera (JB):                1.332
Skew:                           0.050   Prob(JB):                        0.514
Kurtosis:                       2.871   Cond. No.                     8.57e+03
==============================================================================

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC1)
[2] The condition number is large, 8.57e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Bootstrap

In [61]:
```
varSimp = np.var(regSimp_bstr_1)
varCmplx = np.var(regCmplx_bstr_1)
print("Variance of simple model:",round(varSimp,3))
print("Variance of complex model:",round(varCmplx,3))
```

```
Variance of simple model: 0.535
Variance of complex model: 1.833
```

From the results above, the complex model's variance is much higher than simple model.

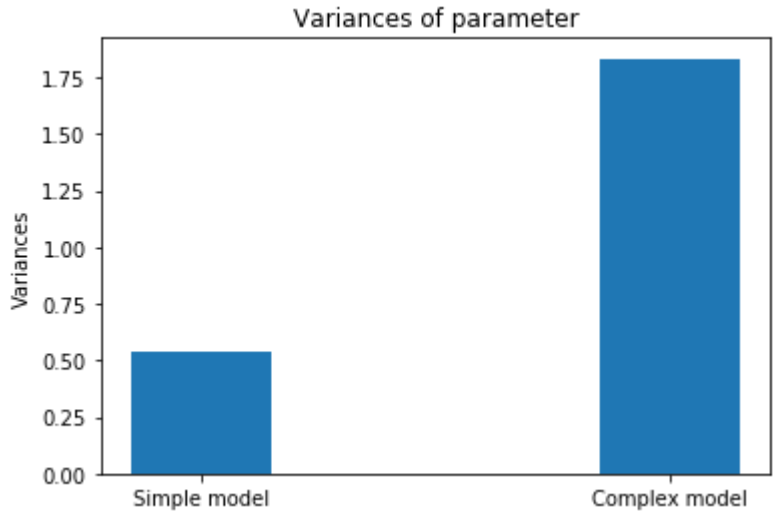
5. (5 points)

Using the same output as in Question 4, depict the comparison of variances graphically.

The below graph shows the variance of each model and the gap between these two variances

In [55]:
```
plt.bar(['Simple model','Complex model'],[varSimp,varCmplx], width = 0.3)
plt.title("Variances of parameter")
plt.ylabel("Variances")
```

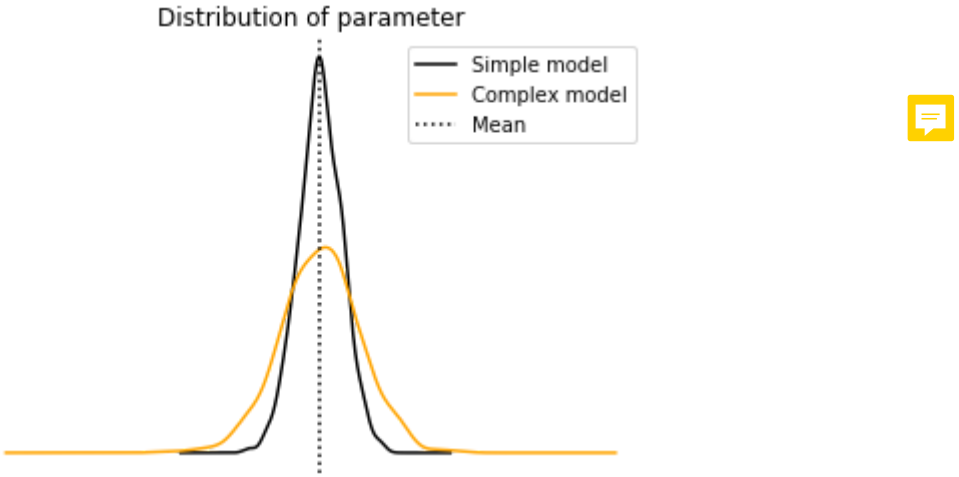
Out[55]:
```
Text(0, 0.5, 'Variances')
```



In below graph, by shifting parameter of complex model toward the simple model's mean, it's clearer to see how the complex model has higher variance than simple model.

In [56]:
```
regCmplx_adj = regCmplx_bstr_1 + (np.mean(regSimp_bstr_1) - np.mean(regCmplx_bstr_1))
plot5 = pd.DataFrame({'Simple model': regSimp_bstr_1,'Complex model': regCmplx_adj})
plot5.plot.kde(color = ['black', 'orange'])
plt.title("Distribution of parameter")
plt.axvline(x=np.mean(regSimp_bstr_1),color='black', ls = ":", label = "Mean")
plt.axis('off')
plt.legend()
```

Out[56]: <matplotlib.legend.Legend at 0x189dc045c50>



6. (10 points)

The data scientist tells you that variance is only part of the story; it is also important to find a model with lower bias. In general, why can we not assess model bias using data only? Despite that, can we still holistically assess the simple versus complex model using data only?

We can not assess model bias using data only since bias is unobserved. We cannot observe outcomes in both scenario at the same time. However, we can assess the simple vs complex model using data by assessing the bias-variance trade-off through mean-squared error.

$$minE[(\hat{\beta} - \beta)^2] = min[E((\hat{\beta} - E(\hat{\beta})^2) + (E(\hat{\beta} - \beta)^2] = min(Variance + Bias)$$

As variance increase, bias has to decrease and vice versa. A simple model might be more biased than a complex model but it might have a lower variance and hence could generalize better than a complex model. Complex model could be closer to the truth but it sacrifice the generalization power because of higher variance. Bias is unobserved, but variance is observable. Hence, by using bias-variance trade off, we can still assess model based on data only.

7. (5 points)

Suppose we return to the simple model. After examining the data for interesting or suspicious patterns, can you identify and justify any potential improvements to this model?

- Summarize data by age and treatment (*data_by_age*); and by channel and treatment (*data_by_chan*). The summarized tables include count of units and difference in time spent for that specific group:

In [40]:
```
#Group data by age
data_by_age = data.groupby(['age','treatment'], as_index=False)[['timespent']].mean()
data_by_age = (data_by_age.join(data.groupby(['age','treatment'], as_index=False).size()
                                .reset_index(name = 'agecounts')['agecounts']))
data_by_age = data_by_age.pivot(index='age', columns='treatment', values=['timespent','agecounts']).reset_ind
ex()
data_by_age.columns = ['age','Y0','Y1','control','treatment']
data_by_age['effect'] = data_by_age['Y1']-data_by_age['Y0']
```

In [27]:
```
data_by_age
```

Out[27]:

	age	Y0	Y1	control	treatment	effect
0	13	300.306872	306.435705	84.0	94.0	6.128833
1	14	283.654762	287.422619	32.0	32.0	3.767857
2	15	273.483640	276.950937	24.0	29.0	3.467297
3	16	268.112969	266.866166	30.0	30.0	-1.246803
4	17	258.715214	258.976891	34.0	34.0	0.261677
5	18	246.986568	251.238254	37.0	32.0	4.251686
6	19	238.225193	241.015767	31.0	37.0	2.790573
7	20	230.803126	233.612719	41.0	26.0	2.809593
8	21	220.568623	222.673286	27.0	32.0	2.104664
9	22	215.811048	214.603499	24.0	20.0	-1.207548
10	23	201.173110	209.228048	26.0	26.0	8.054938
11	24	197.334104	194.276248	32.0	24.0	-3.057856
12	25	183.107645	189.924786	31.0	31.0	6.817141
13	26	174.604331	180.342163	30.0	32.0	5.737832
14	27	169.061046	172.267127	28.0	28.0	3.206081
15	28	160.172207	158.749645	26.0	25.0	-1.422562
16	29	150.697733	150.283256	31.0	40.0	-0.414477
17	30	141.281750	145.189754	32.0	28.0	3.908004

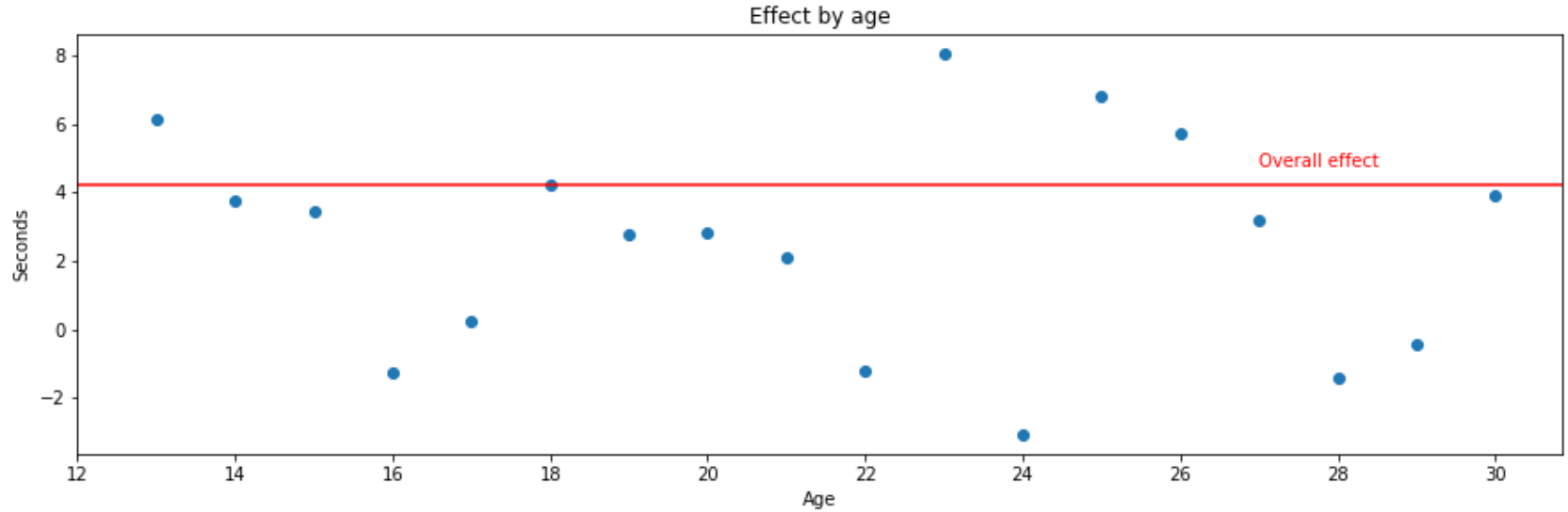
- Visualize the trend of effect by age and by channel; and the proportion of each age and channel in treatment and control group:

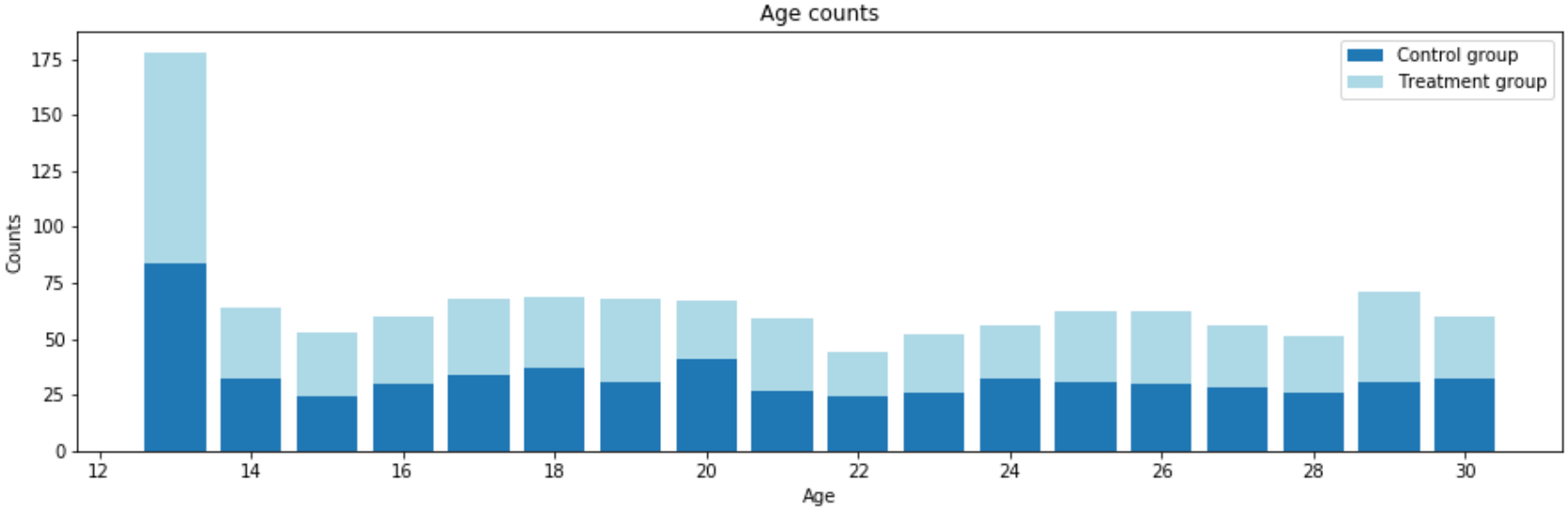
In [59]:
```
#Plot effect by age and by number of channels, distribution of age and number of channels in the sample
f, ax = plt.subplots(2,1,figsize=(15,10))
plt.subplots_adjust(hspace = 0.3)

#Effect by age
ax[0].scatter(data_by_age['age'],data_by_age['effect'])
ax[0].set_title('Effect by age')
ax[0].set(xlabel="Age",ylabel="Seconds")
ax[0].axhline(y=te,color='r')
ax[0].text(27,te+0.5,'Overall effect', color = 'r')
ax[0].set_xticks(np.arange(12, 31, step=2))

#Age counts
ax[1].bar(data_by_age['age'],data_by_age['control'], label = "Control group")
ax[1].bar(data_by_age['age'],data_by_age['treatment'], label = "Treatment group",
            bottom=data_by_age['control'], color = 'lightblue')
ax[1].set_title('Age counts')
ax[1].set(xlabel="Age",ylabel="Counts")
ax[1].legend()
ax[1].set_xticks(np.arange(12, 31, step=2))

plt.show()
```


Effect by age
Seconds
Age
Overall effect
8
6
4
2
0
−2
12
14
16
18
20
22
24
26
28
30


Age counts
Counts
Age
Control group
Treatment group
175
150
125
100
75
50
25
0
12
14
16
18
20
22
24
26
28
30

- Observation from the graph:
  - Effect varies wildly between different age groups (See graph *Effect by age*).
  - Except age 13, all other age groups are not very different in terms of total units in experiment. The proportion between treatment and control groups across all age groups is also quite consistent (See graph *Age counts*).
  - The presence of age 13 in the population is much more noticeable than all other ages. Younger users might be more attracted to app like TikTok. With or without the new feature, they already spent much more time on the app than other users. If this group has certain characteristics that are distinctly different from the rest, the effect of these characteristics will be projected on the overall effect.
  - In general, there is relatively gradual change in effect between age groups. However, from age 22-25, there are unusual leaps among these ages. There should not be such a dramatic change between these ages since they are very closed to each other. There could be noise in this subset of data. Or units in these age groups might have special characteristics.

From the observation above, I would consider blocking age 13 and ages 23-25 from the rest.