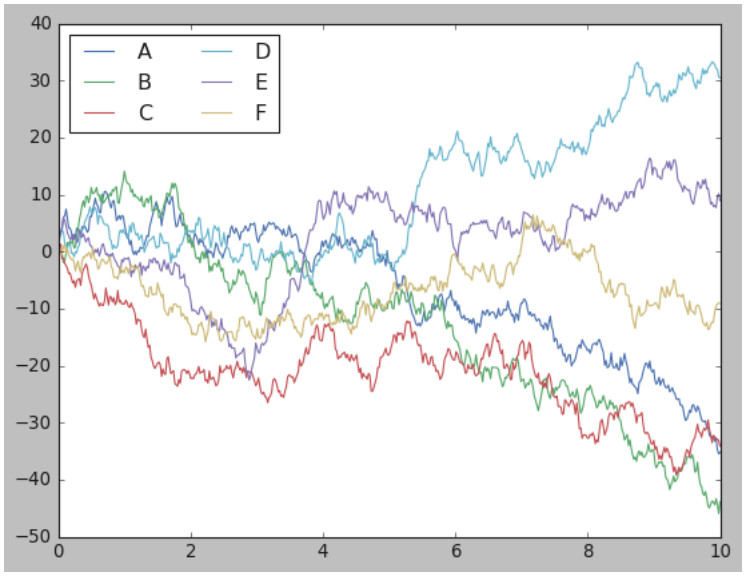
In [93]:
```
import matplotlib.pyplot as plt
plt.style.use("classic")
%matplotlib inline
import numpy as np
import pandas as pd
```

In [94]:
```
rng=np.random.RandomState(0)
x=np.linspace(0,10,500)
y=np.cumsum(rng.randn(500,6),0)
plt.plot(x,y)
plt.legend("ABCDEF",ncol=2,loc="upper left")
```

Out[94]: <matplotlib.legend.Legend at 0x272162c05e0>

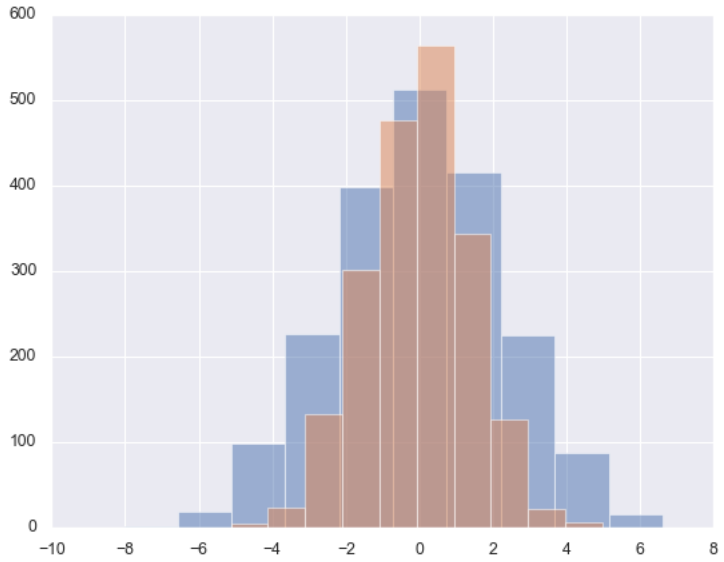


In [95]:
```
import seaborn as sns
sns.set()
```

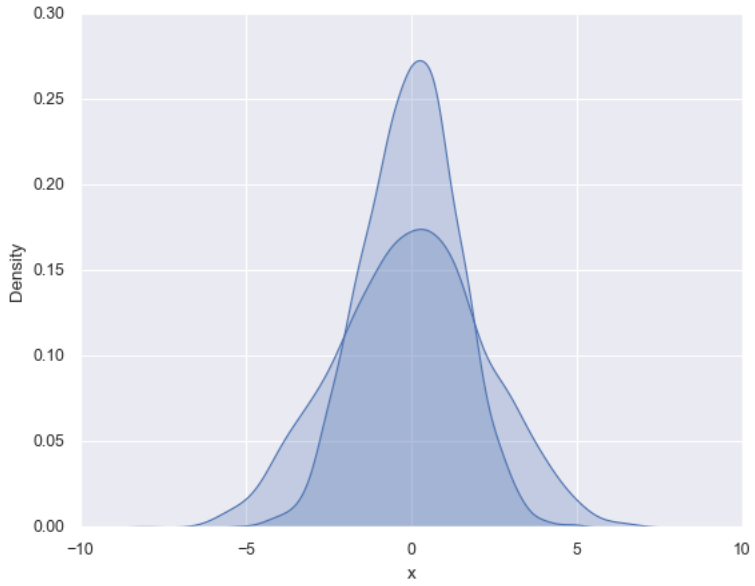
In [96]:
```
#plotting the same graph using seaborn
plt.plot(x,y)
plt.legend("ABCDEF",ncol=2,loc="upper left")
```

Out[96]: <matplotlib.legend.Legend at 0x27214d15ee0>

In [97]:
```
data=np.random.multivariate_normal([0,0],[[5, 2], [2, 2]],size=2000)
data=pd.DataFrame(data,columns=["x","y"])
for col in "xy":
    plt.hist(data[col],alpha=0.5)
```



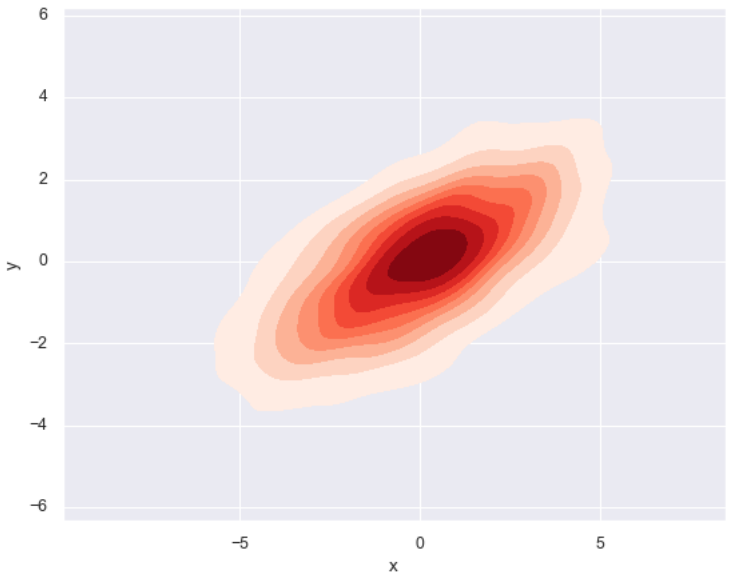
In [98]:
```
#to get a smooth distribution curve we kernal density estimation in seaborn
for col in "xy":
    sns.kdeplot(data[col],shade=True)
```



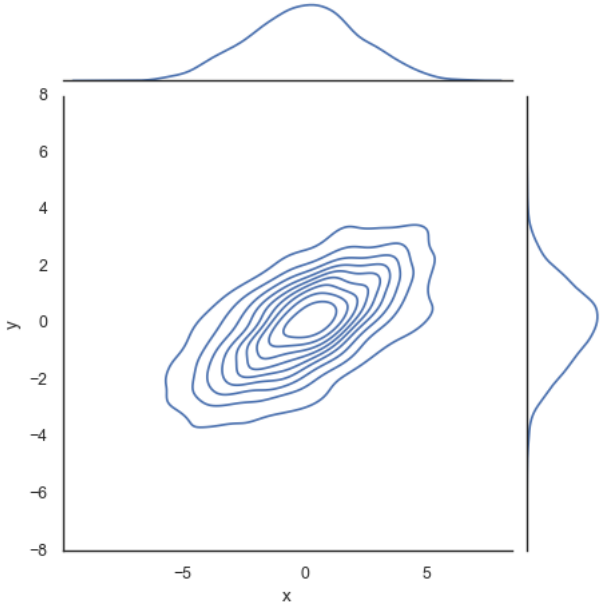
In [99]:
```
#histograms and kde can be combined using distplot however displot is deprecated now
```

In [100]:
```
#passing data[columns] in kdeplot we get 2D data
sns.kdeplot(x=data["x"],y=data["y"],shade=True,cmap="Reds")
```

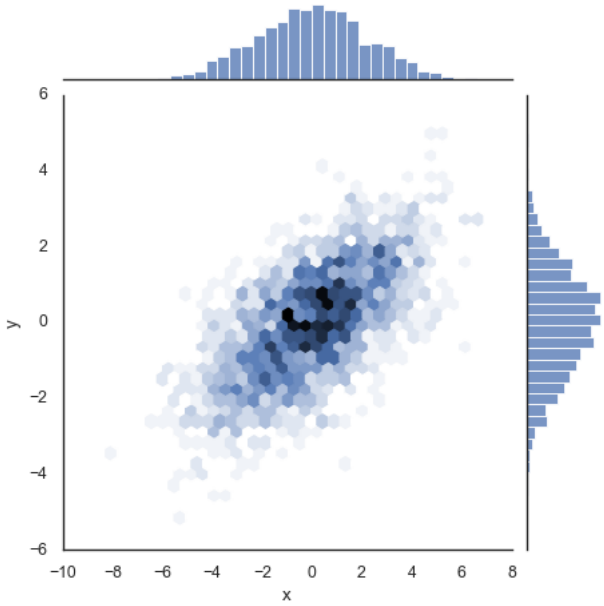
Out[100]: <AxesSubplot:xlabel='x', ylabel='y'>



In [101]:
```
with sns.axes_style("white"):#jointplot
    sns.jointplot(x="x",y="y",data=data,kind="kde")
```



In [102]:
```
with sns.axes_style("white"):#hexagonally based histogram
    sns.jointplot(x="x",y="y",data=data,kind="hex")
```



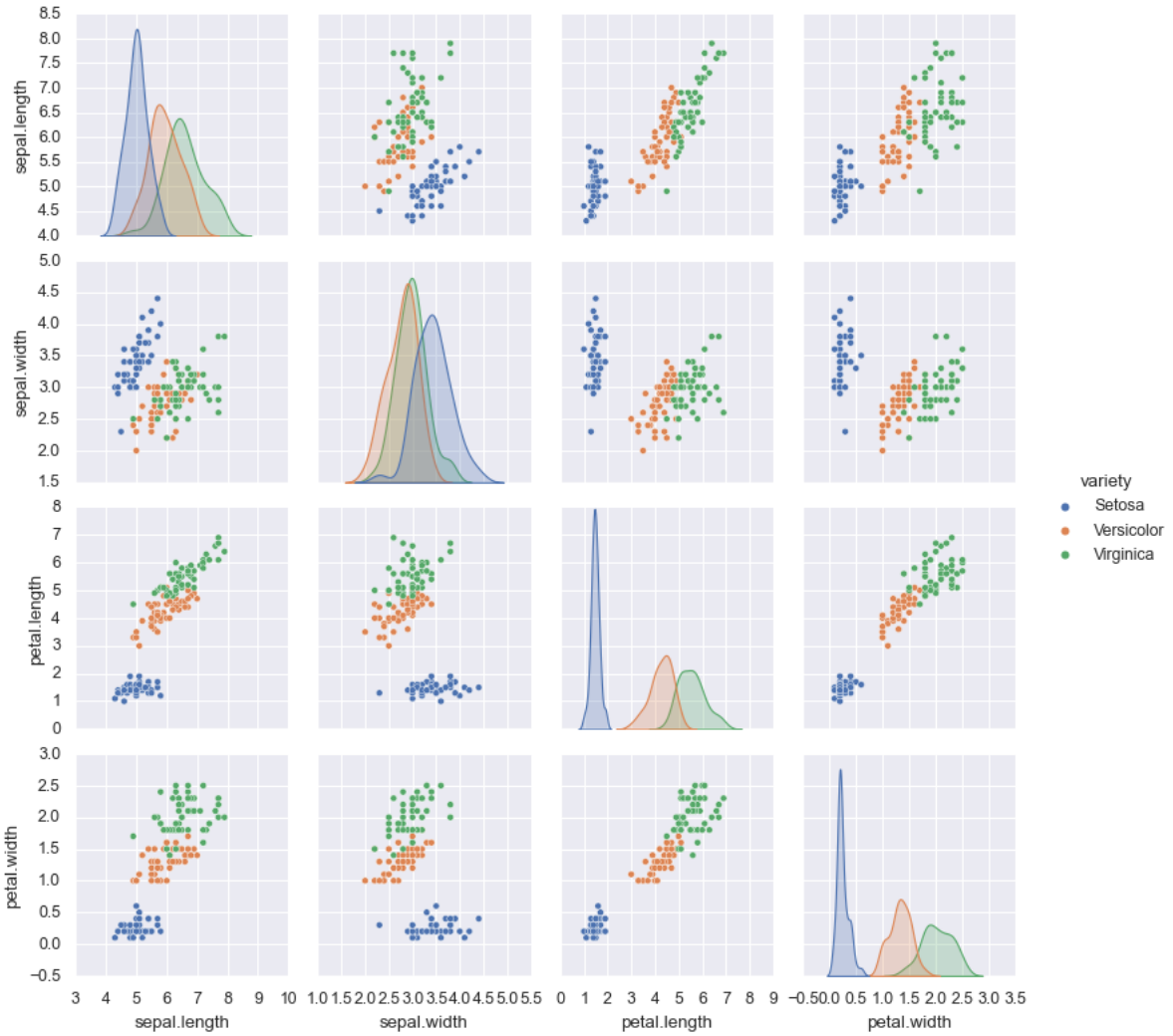
In [103]:
```
iris=pd.read_csv("Iris.csv")#iris dataset has measurements of petals and sepals from iris species
iris.head()
```

Out[103]:

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

In [104]:
```
sns.pairplot(iris,hue="variety",height=2.5)#hue is given as a clumn name to mark colors
```

Out[104]:
```
<seaborn.axisgrid.PairGrid at 0x27217aa70a0>
```



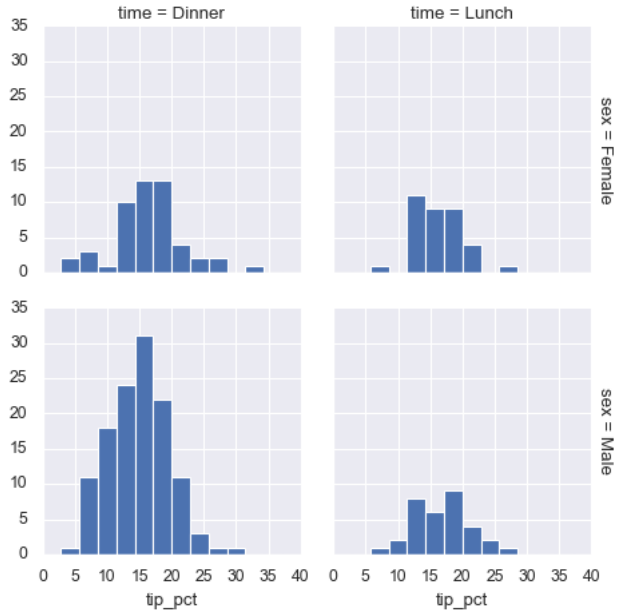
In [105]:
```
tips=pd.read_csv("tips.csv")
tips.head()
```

Out[105]:

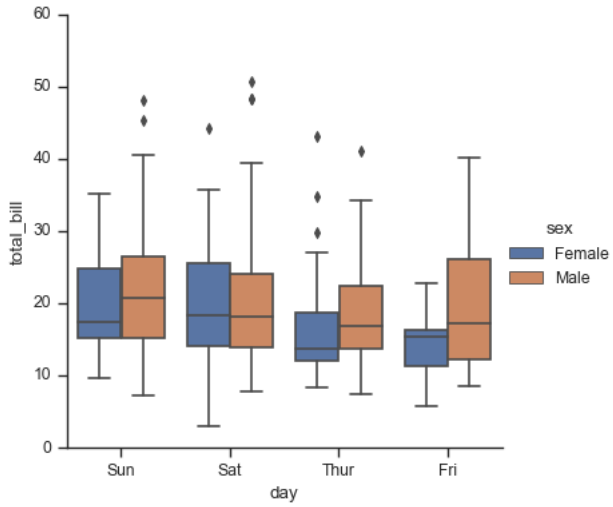
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [106]:
```
#faceted histograms are histograms plotted on subsets of data
tips['tip_pct']=100*(tips["tip"]/tips["total_bill"])#adding tips percentage column
grid=sns.FacetGrid(tips,row="sex",col="time",margin_titles=True)#assigning row as sex and column and time
grid.map(plt.hist,"tip_pct",bins=np.linspace(0, 40, 15))
```

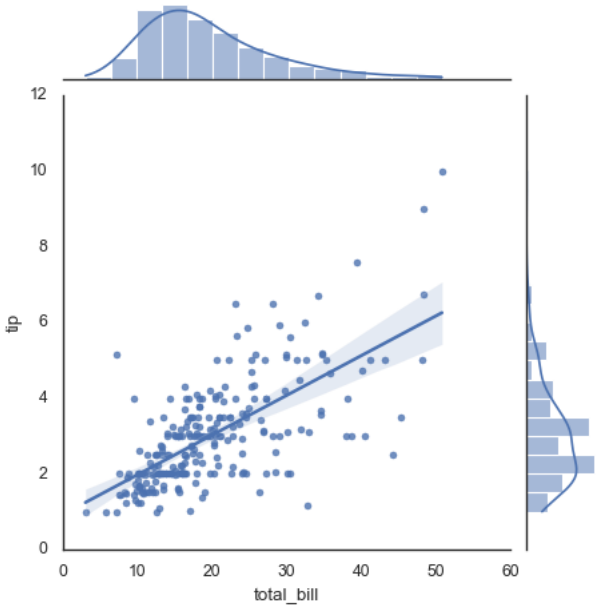
Out[106]: <seaborn.axisgrid.FacetGrid at 0x272180c0fd0>



In [107]:
```
with sns.axes_style(style="ticks"):
    sns.catplot(x="day",y="total_bill",hue="sex",data=tips,kind="box")#x axis value,y axis value,color coded info(hue),data
#factorplot has been renamed to catplot
```



In [108]:
```
#regression in joint plots
with sns.axes_style("white"):#hexagonally based histogram
    sns.jointplot(x="total_bill",y="tip",data=tips,kind="reg")
```

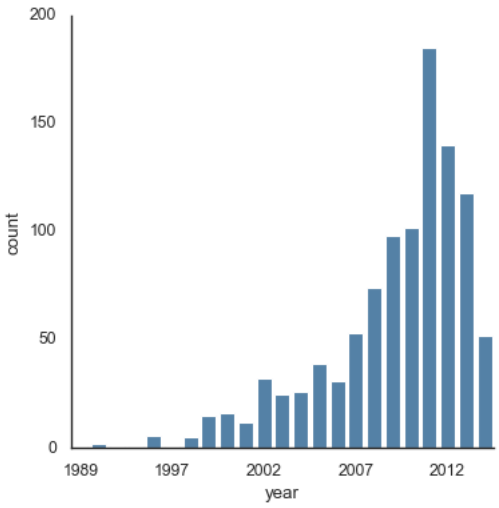


In [109]:
```
planets=sns.load_dataset("planets")
planets.head()
```

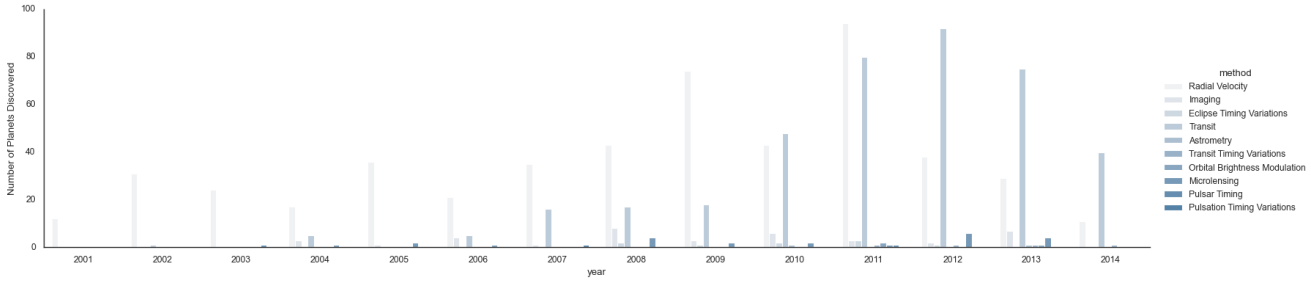
Out[109]:

	method	number	orbital_period	mass	distance	year
0	Radial Velocity	1	269.300	7.10	77.40	2006
1	Radial Velocity	1	874.774	2.21	56.95	2008
2	Radial Velocity	1	763.000	2.60	19.84	2011
3	Radial Velocity	1	326.030	19.40	110.62	2007
4	Radial Velocity	1	516.220	10.50	119.47	2009

In [110]:
```
#bar plots in seaborn
with sns.axes_style('white'):
    g=sns.catplot(x="year",data=planets,kind="count",color="steelblue")
    g.set_xticklabels(step=5)
```



In [111]:
```
with sns.axes_style('white'):
    g=sns.catplot(x="year",data=planets,aspect=4.0,kind="count",color="steelblue",hue="method",order=range(2001, 2015))
    g.set_ylabels("Number of Planets Discovered")
```



In [128]:
```
data = pd.read_csv("marathon-data.csv")
data.head()
```

Out[128]:

	age	gender	split	final
0	33	M	01:05:38	02:08:51
1	32	M	01:06:26	02:09:28
2	31	M	01:06:49	02:10:42
3	38	M	01:06:16	02:13:45
4	31	M	01:06:32	02:13:59

In [129]:
```
data.dtypes#time is loaded as string hence we have to convert to datetime format
```

Out[129]:
```
age        int64
gender    object
split     object
final     object
dtype: object
```

In [130]:
```
def convert_time(s):
    h, m, s = map(int, s.split(':'))
    return pd.Timedelta(hours=h, minutes=m, seconds=s)
data = pd.read_csv('marathon-data.csv',converters={'split':convert_time, 'final':convert_time})
data.head()
```

Out[130]:

	age	gender	split	final
0	33	M	0 days 01:05:38	0 days 02:08:51
1	32	M	0 days 01:06:26	0 days 02:09:28
2	31	M	0 days 01:06:49	0 days 02:10:42
3	38	M	0 days 01:06:16	0 days 02:13:45
4	31	M	0 days 01:06:32	0 days 02:13:59

In [131]:
```
data.dtypes
```

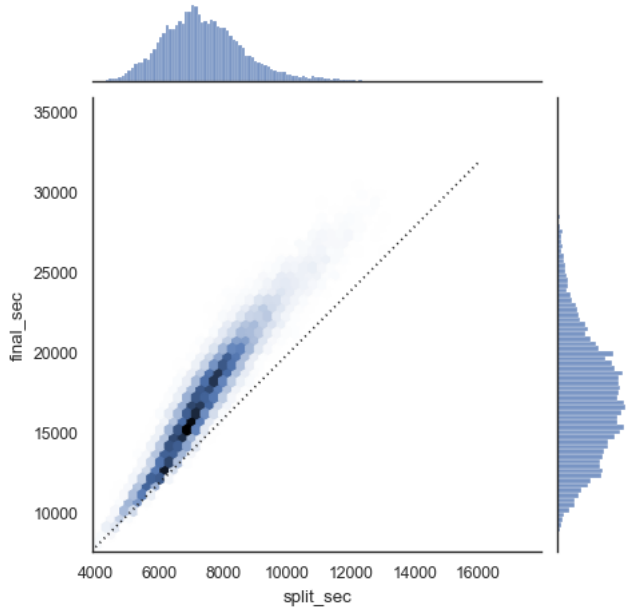
Out[131]:
```
age                int64
gender            object
split     timedelta64[ns]
final     timedelta64[ns]
dtype: object
```

In [139]:
```
data["split_sec"]=data["split"]/pd.Timedelta(seconds=1)#creating new column split second
data["final_sec"]=data["final"]/pd.Timedelta(seconds=1)#creating new column final second
data.head()
```

Out[139]:

	age	gender	split	final	split_sec	final_sec
0	33	M	0 days 01:05:38	0 days 02:08:51	3938.0	7731.0
1	32	M	0 days 01:06:26	0 days 02:09:28	3986.0	7768.0
2	31	M	0 days 01:06:49	0 days 02:10:42	4009.0	7842.0
3	38	M	0 days 01:06:16	0 days 02:13:45	3976.0	8025.0
4	31	M	0 days 01:06:32	0 days 02:13:59	3992.0	8039.0

In [144]:
```
with sns.axes_style("white"):
    g=sns.jointplot(x="split_sec",y="final_sec",data=data,kind="hex")
    plt.plot(np.linspace(4000, 16000),np.linspace(8000, 32000), ':k')
```

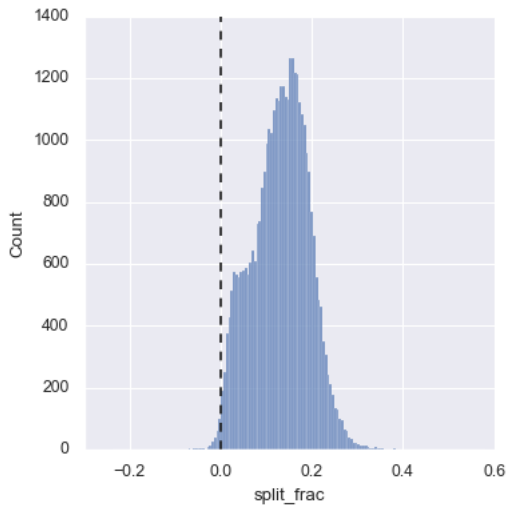


In [146]:
```
data["split_frac"] =1 -2 * data["split_sec"] / data["final_sec"]#formula for calculating split
#positive split means fast at first half of race and slowing down negative split is the reverse
data.head()
```

Out[146]:

	age	gender	split	final	split_sec	final_sec	split_frac
0	33	M	0 days 01:05:38	0 days 02:08:51	3938.0	7731.0	-0.018756
1	32	M	0 days 01:06:26	0 days 02:09:28	3986.0	7768.0	-0.026262
2	31	M	0 days 01:06:49	0 days 02:10:42	4009.0	7842.0	-0.022443
3	38	M	0 days 01:06:16	0 days 02:13:45	3976.0	8025.0	0.009097
4	31	M	0 days 01:06:32	0 days 02:13:59	3992.0	8039.0	0.006842

In [148]:
```
sns.displot(data["split_frac"], kde=False);#kde false implies that kde plot is invisible
plt.axvline(0,color="k",linestyle="--")#axvline is used for plotting a vertical line
```

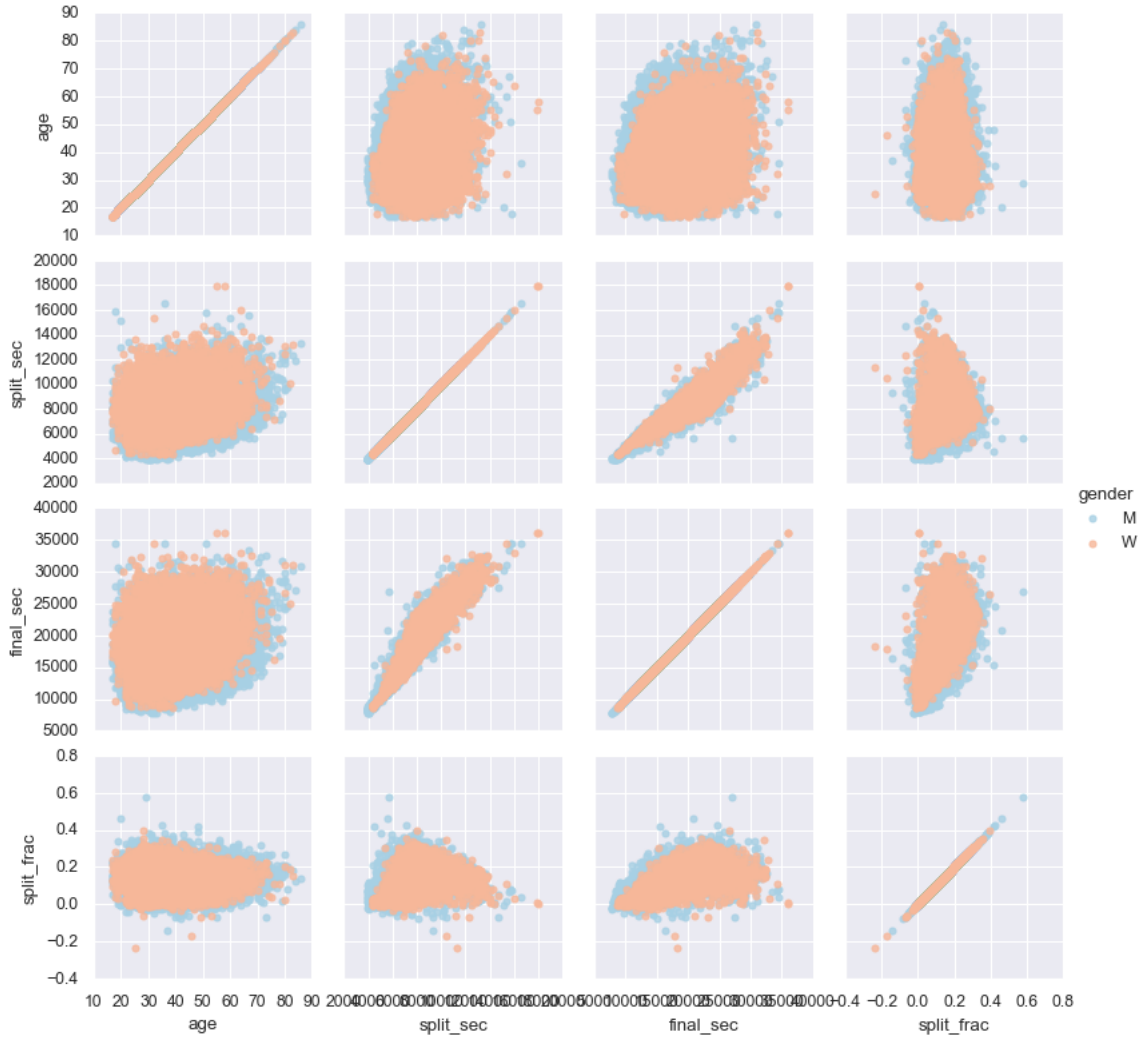


In [149]:
```
sum(data.split_frac<0)#number of people with negative split
```

Out[149]: 251

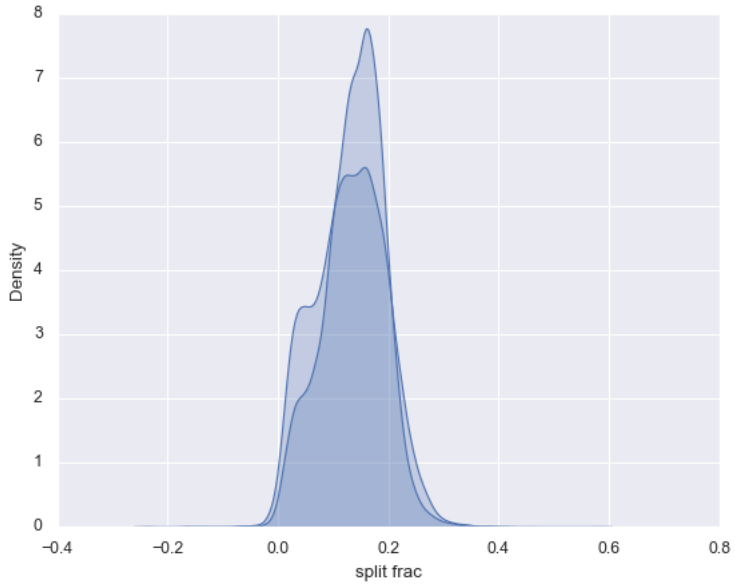
In [150]:
```
g=sns.PairGrid(data,vars=["age","split_sec","final_sec","split_frac"],hue="gender",palette="RdBu_r")
g.map(plt.scatter, alpha=0.8)
g.add_legend()
```

Out[150]: <seaborn.axisgrid.PairGrid at 0x272567164c0>



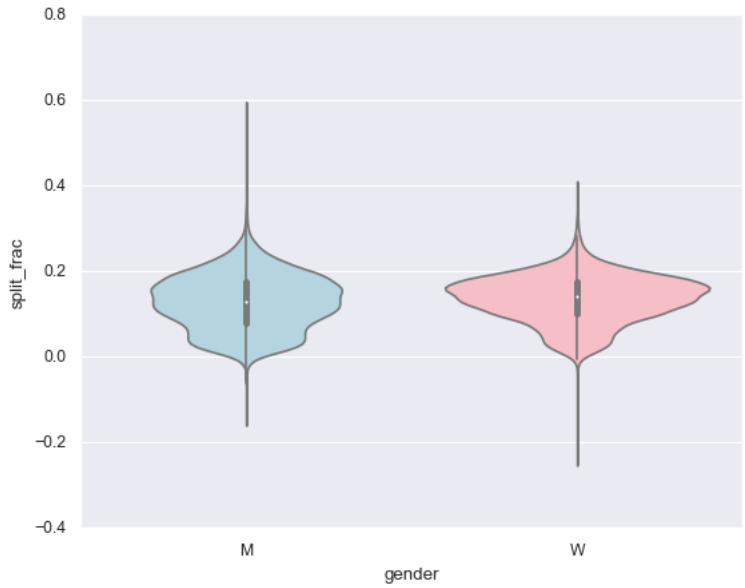
In [153]:
```
#comparing split fraction of men and women
sns.kdeplot(data.split_frac[data.gender=="M"],label="men",shade=True)
sns.kdeplot(data.split_frac[data.gender=="W"],label="women",shade=True)
plt.xlabel("split frac")
```

Out[153]:
```
Text(0.5, 0, 'split frac')
```



In [155]:
```
#violin plot comaparison
sns.violinplot(x="gender",y="split_frac",data=data,palette=["lightblue","lightpink"])
```

Out[155]:
```
<AxesSubplot:xlabel='gender', ylabel='split_frac'>
```

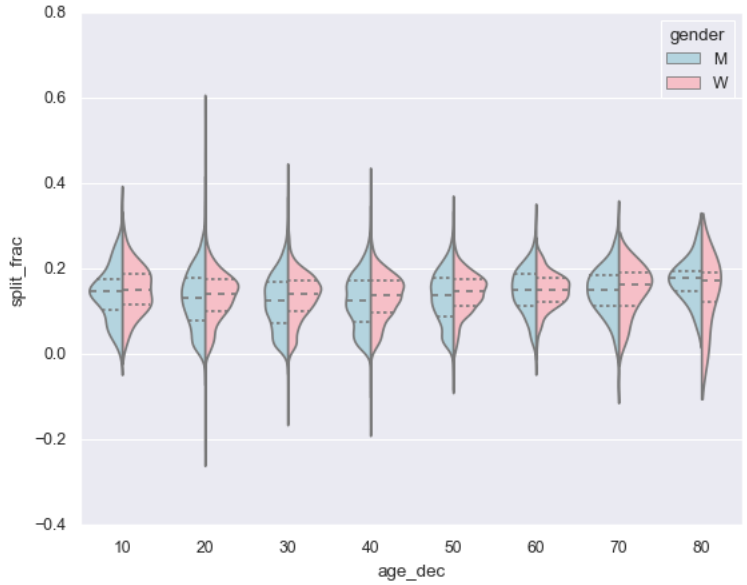


In [156]:
```
data["age_dec"]=data.age.map(lambda age:10*(age//10))#adding new column age decade
data.head()
```

Out[156]:

	age	gender	split	final	split_sec	final_sec	split_frac	age_dec
0	33	M	0 days 01:05:38	0 days 02:08:51	3938.0	7731.0	-0.018756	30
1	32	M	0 days 01:06:26	0 days 02:09:28	3986.0	7768.0	-0.026262	30
2	31	M	0 days 01:06:49	0 days 02:10:42	4009.0	7842.0	-0.022443	30
3	38	M	0 days 01:06:16	0 days 02:13:45	3976.0	8025.0	0.009097	30
4	31	M	0 days 01:06:32	0 days 02:13:59	3992.0	8039.0	0.006842	30

In [157]:
```
with sns.axes_style(style=None):
    sns.violinplot(x="age_dec",y="split_frac",hue="gender",data=data,split=True,inner="quartile",
                   palette=["lightblue","lightpink"]);
```



In [164]:
```
#linear regression
g = sns.lmplot(x='final_sec',y='split_frac', col='gender', data=data,
               markers=".", scatter_kws=dict(color='c'))
g.map(plt.axhline,y=0.1,color="k",ls=":")
```

Out[164]: <seaborn.axisgrid.FacetGrid at 0x272582ce100>