0.0.1 Question 2c: Verify Outcome

Did the candidate win or lose the election? Verify with election outcome.

```
In [16]: election_sub[election_sub["candidate"]=='Sharice Davids']
```

```
Out[16]:         year office state  district election_date forecast_date forecast_type  \
         204324  2018  House    KS       3.0    2018-11-06    2018-11-06       classic
         455289  2018  House    KS       3.0    2018-11-06    2018-08-11       classic

                party       candidate  projected_voteshare  actual_voteshare  probwin  \
         204324     D  Sharice Davids             51.85115               NaN  0.84994
         455289     D  Sharice Davids             44.84660               NaN  0.19566

                 probwin_outcome             bin
         204324                1  (0.842, 0.895]
         455289                1  (0.158, 0.211]
```

```
In [17]: election_sub[election_sub["candidate"]=='Adam B. Schiff']
```

```
Out[17]:         year office state  district election_date forecast_date forecast_type  \
         205075  2018  House    CA      28.0    2018-11-06    2018-11-06       classic
         456037  2018  House    CA      28.0    2018-11-06    2018-08-11       classic

                party       candidate  projected_voteshare  actual_voteshare  probwin  \
         205075     D  Adam B. Schiff             81.81078               NaN      1.0
         456037     D  Adam B. Schiff             80.84081               NaN      1.0

                 probwin_outcome           bin
         205075                1  (0.947, 1.0]
         456037                1  (0.947, 1.0]
```

The rising candidate, Sharice Davids, won the election.

The falling candidate, Adam B. Schiff, won the election.

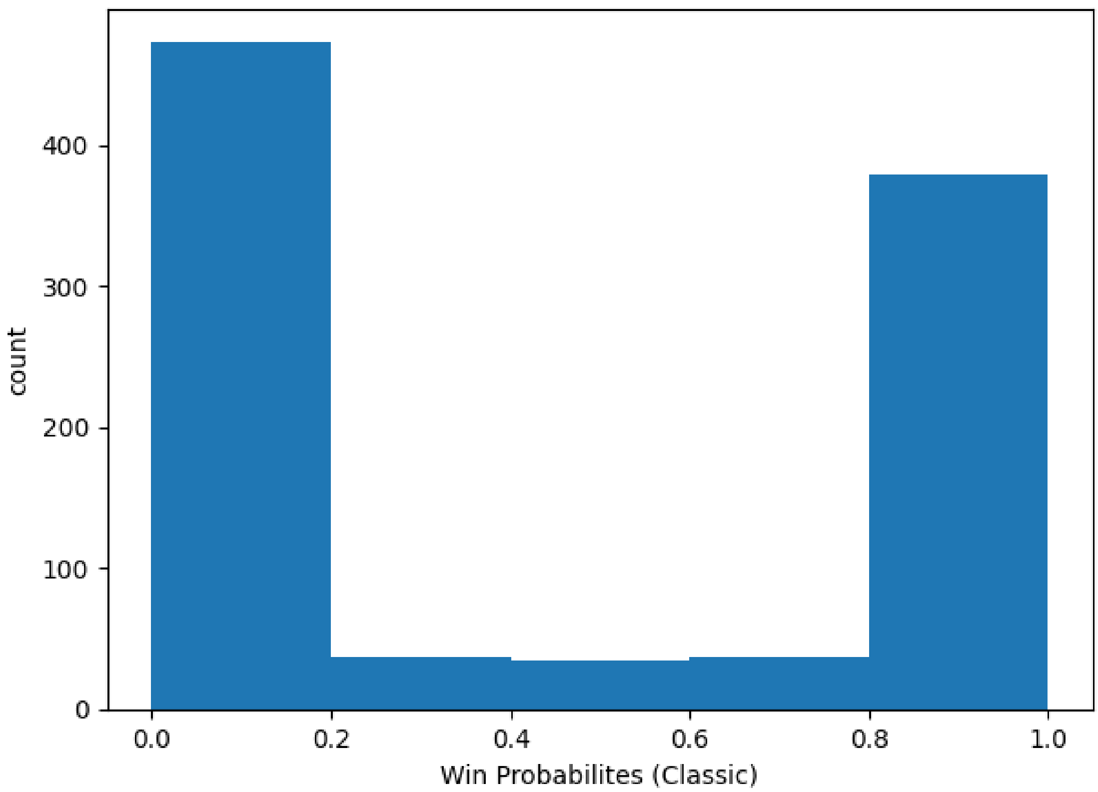
0.0.2 Question 3a: Prediction Histogram

Make a histogram showing the predicted win probabilities *on the morning of the election.* Again, restrict yourself to only the `classic` predictions.

In [18]:
```
classic_preds=election_sub[(election_sub['forecast_date']=='2018-11-06') & (election_sub['fore
winprobs_classic=classic_preds['probwin']

import matplotlib.pyplot as plt

plt.figure(figsize=(7,5))
plt.hist(winprobs_classic, bins=5)
plt.xlabel('Win Probabilites (Classic)')
plt.ylabel('count')
plt.show()
```



0.0.3 Question 3b: Prediction difficulty

Are most house elections easy to forecast or hard to forecast? State your reasoning.

Most house elections are easy to forecast because a majority of the win probabilities are either really high or really low, indicating a high amount of confidence in a certain outcome.

If the number of win probabilities with a value of 0.5 was the highest, then I would say that the election is difficult to predict because there is no certaintanty for either outcome.

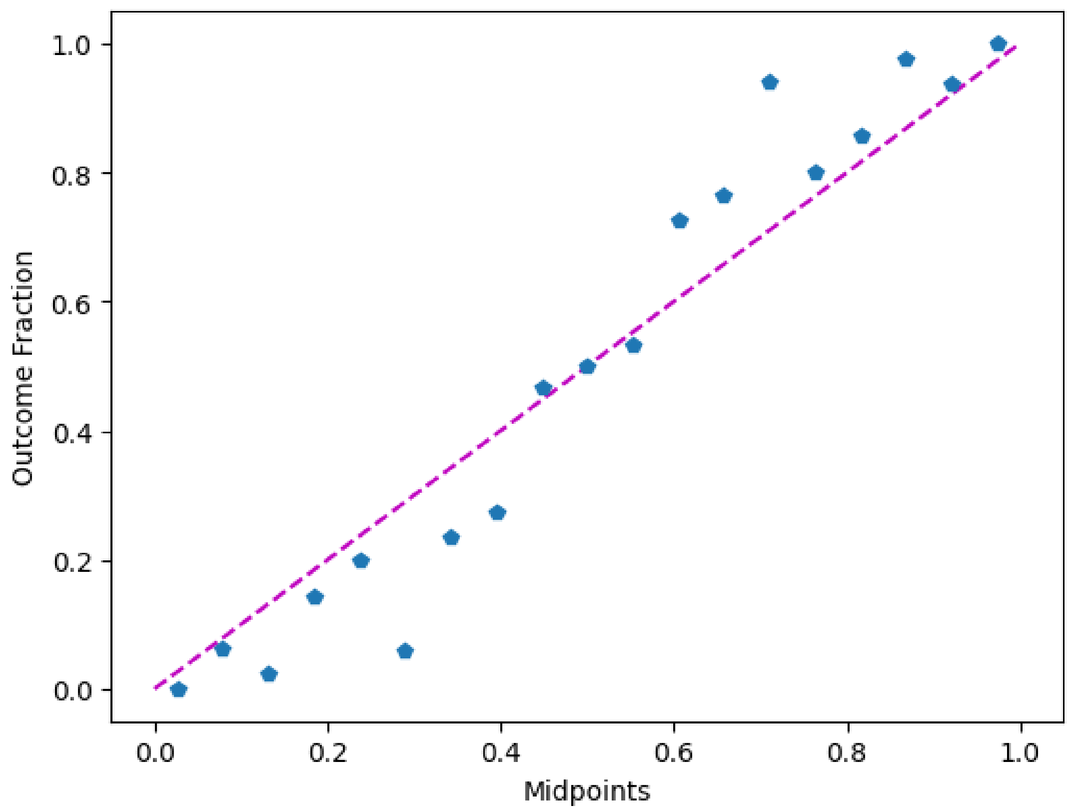
0.0.4 Question 4c: Visualize Results

Now make a scatterplot using `midpoints` as the x variable and `fraction_outcome` as the y variable. Draw a dashed line from `[0,0]` to `[1,1]` to mark the line y=x.

```
In [23]: # magic for showing figures inline
         %matplotlib inline
         import matplotlib.pyplot as plt

         plt.plot([0,1], 'm--')
         plt.plot(midpoints, fraction_outcome, 'p')
         plt.xlabel('Midpoints')
         plt.ylabel('Outcome Fraction')
```

```
Out[23]: Text(0, 0.5, 'Outcome Fraction')
```

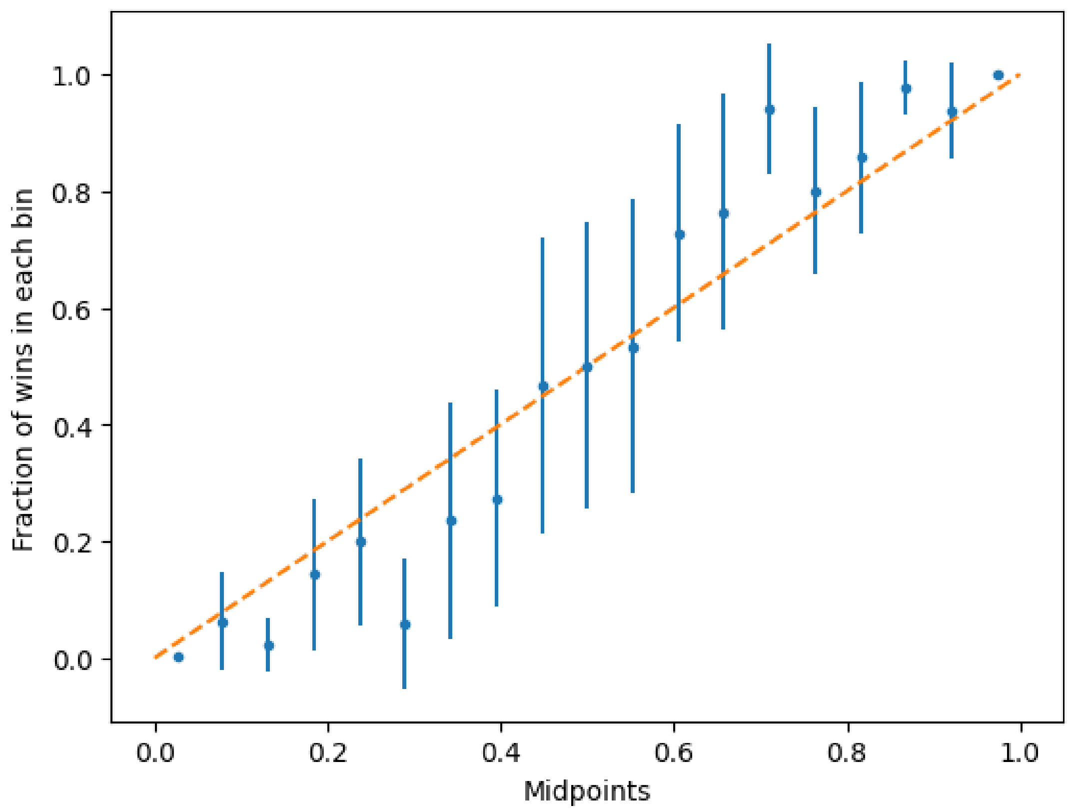


0.0.5 Question 5b: Visualize Error Bars 1

Use `plt.errorbar` to create a new plot with error bars associated with the actual fraction of wins in each bin. Again add a dashed y=x line. Set the argument `fmt='.'` to create a scatterplot with errorbars.

```
In [26]: # Plotting code below
         plt.errorbar(midpoints, election_agg['mean'], yerr=election_agg['err'], fmt='.')
         plt.plot([0, 1], [0, 1], '--')
         plt.xlabel('Midpoints')
         plt.ylabel('Fraction of wins in each bin')
```

```
Out[26]: Text(0, 0.5, 'Fraction of wins in each bin')
```



0.0.6 Question 5d: Understanding Confidence Intervals

Are the 95% confidence intervals generally larger or smaller for more confident predictions (e.g. the predictions closer to 0 or 1). What are the factors that determine the length of the confidence intervals?

The 95% confidence intervals are generally smaller for more confident predictions. Sample size, standard deviation (variation in the sample), and the confidence level are all factors that determine the length of the confidence intervals.

0.0.7 (PSTAT 234) Question 5f. Visualize Error Bars 2

By now, we have a distribution of success probabilities saved in `bootstrap_election_agg`. We can compute empirical error bars from 2.5% and 97.5% quantiles. Write function named `bootstrap_error_bars` that can be used to calculate the following columns:

- `mean`: mean of probabilities of success
- `err_low`: low point of the error bars
- `err_high`: high point of the error bars

Funtion `bootstrap_error_bars` is to be called by using `bootstrap_election_100_agg.apply(bootstrap_error_bars, ...)`.

In [ ]:
```
def bootstrap_error_bars(x):
    out = pd.Series([x.mean(), x.mean()-x.quantile(0.025), x.quantile(0.975)-x.mean()],
                    index=['mean', 'err_low', 'err_high'])
    return(out)
```

0.0.8 (PSTAT 234) Question 5g: Interpreting the Results

Are the 95% confidence intervals generally larger or smaller for more confident predictions (e.g. the predictions closer to 0 or 1). What are the factors that determine the length of the error bars?

Compare and contrast model-based error bars and empirically obtained error bars. What are the advantages and disadvantages of these two approaches?

Type your answer here, replacing this text.