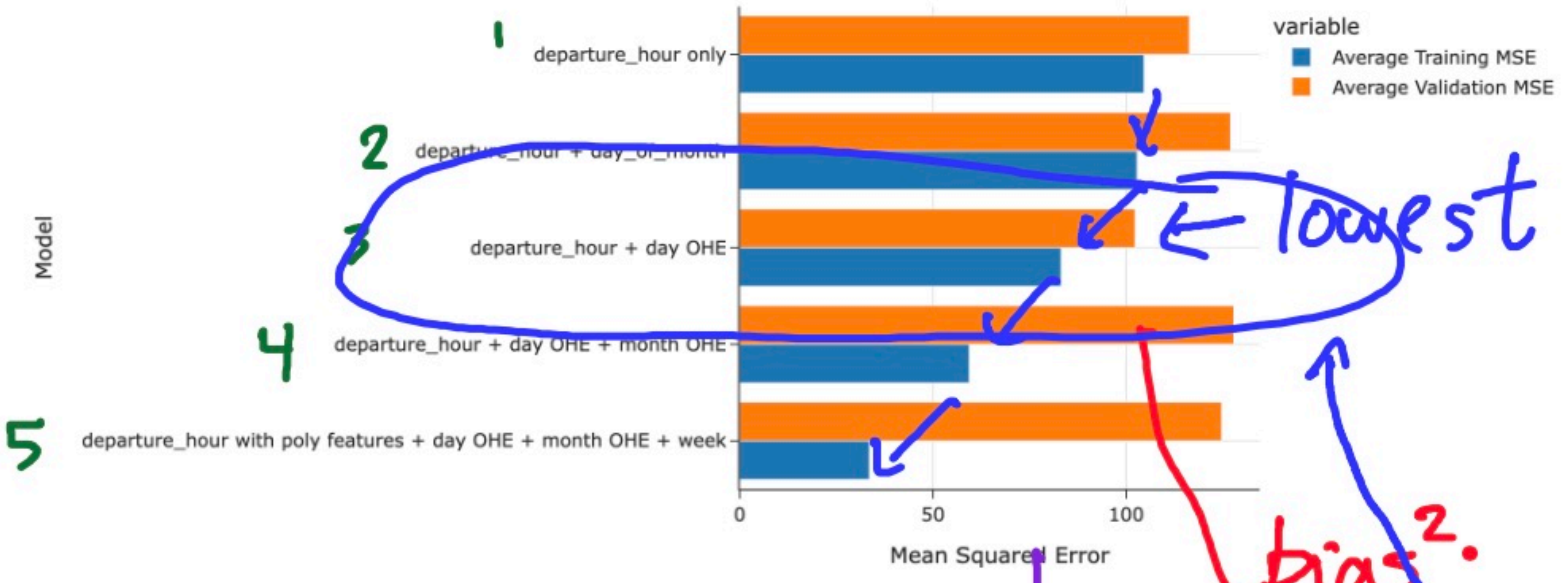
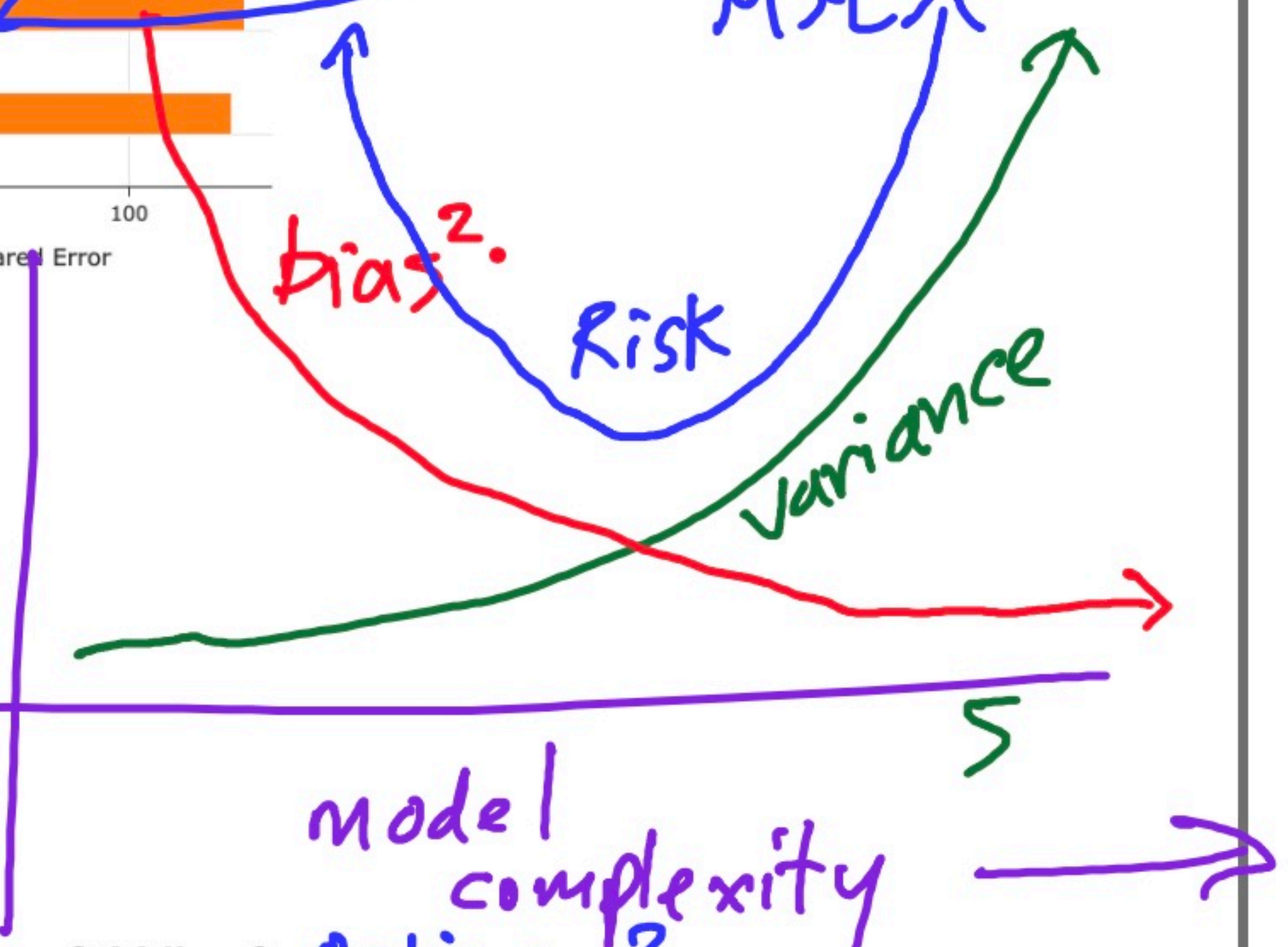
- Last class, we used k-fold cross-validation to choose between the following five models that predict commute time in `'minutes'`.





- Which model has the highest **model bias**?

See the annotated slides for the answer.

Option 1: simplest highest training MSE

- Which model has the highest **model variance**?

Option 5: lowest training MSE

- Which model is most likely to perform best in practice? Why? Option 3

Ridge regression

- **Idea**: In addition to just minimizing mean squared error, what if we could **also** try and prevent large parameter values?
  Maybe this would lead to less overfitting!

new objective function

- **Regularization** is the act of adding a penalty on the norm of the parameter vector, $\vec{w}$, to the objective function.

$$R_{\text{ridge}}(\vec{w}) = \frac{1}{n}\|\vec{y} - X\vec{w}\|^2 + \underbrace{\lambda \sum_{j=1}^{d} w_j^2}_{\substack{\text{regularization} \\ \text{penalty}}}$$

don't regularize intercept

- Linear regression with L_2 regularization – as shown above – is called **ridge regression**.
  You'll explore the reason why in Homework 9!

$\lambda \|\vec{w}\|_2^2$

Activity

The objective function we minimize to find $\vec{w}^*_{\text{ridge}}$ in **ridge regression** is:

$$R_{\text{ridge}}(\vec{w}) = \frac{1}{n}\|\vec{y} - X\vec{w}\|^2 + \cancel{\lambda \sum_{j=1}^{d} w_j^2}$$

λ is a **hyperparameter**, which we choose through cross-validation. Discuss the following points with those near you:

- What if we pick $\lambda = 0$ – what is $\vec{w}^*_{\text{ridge}}$ then?
- What happens to $\vec{w}^*_{\text{ridge}}$ as $\lambda \to \infty$?
- Can λ be negative?

$\vec{w}^*_{ridge} = (X^TX)^{-1}X^T\vec{y}$

like before

if $(X^TX)^{-1}$ exists

Activity

The objective function we minimize to find $\vec{w}_{\text{ridge}}^*$ in **ridge regression** is:

$$R_{\text{ridge}}(\vec{w}) = \frac{1}{n}\|\vec{y} - X\vec{w}\|^2 + \lambda \sum_{j=1}^{d} w_j^2$$

10000000

λ is a **hyperparameter**, which we choose through cross-validation. Discuss the following points with those near you:

- What if we pick $\lambda = 0$ – what is $\vec{w}_{\text{ridge}}^*$ then?
- What happens to $\vec{w}_{\text{ridge}}^*$ as $\lambda \to \infty$?
- Can λ be negative?

$\vec{w}_{\text{ridge}} \to \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ (except for intercept)

no!!

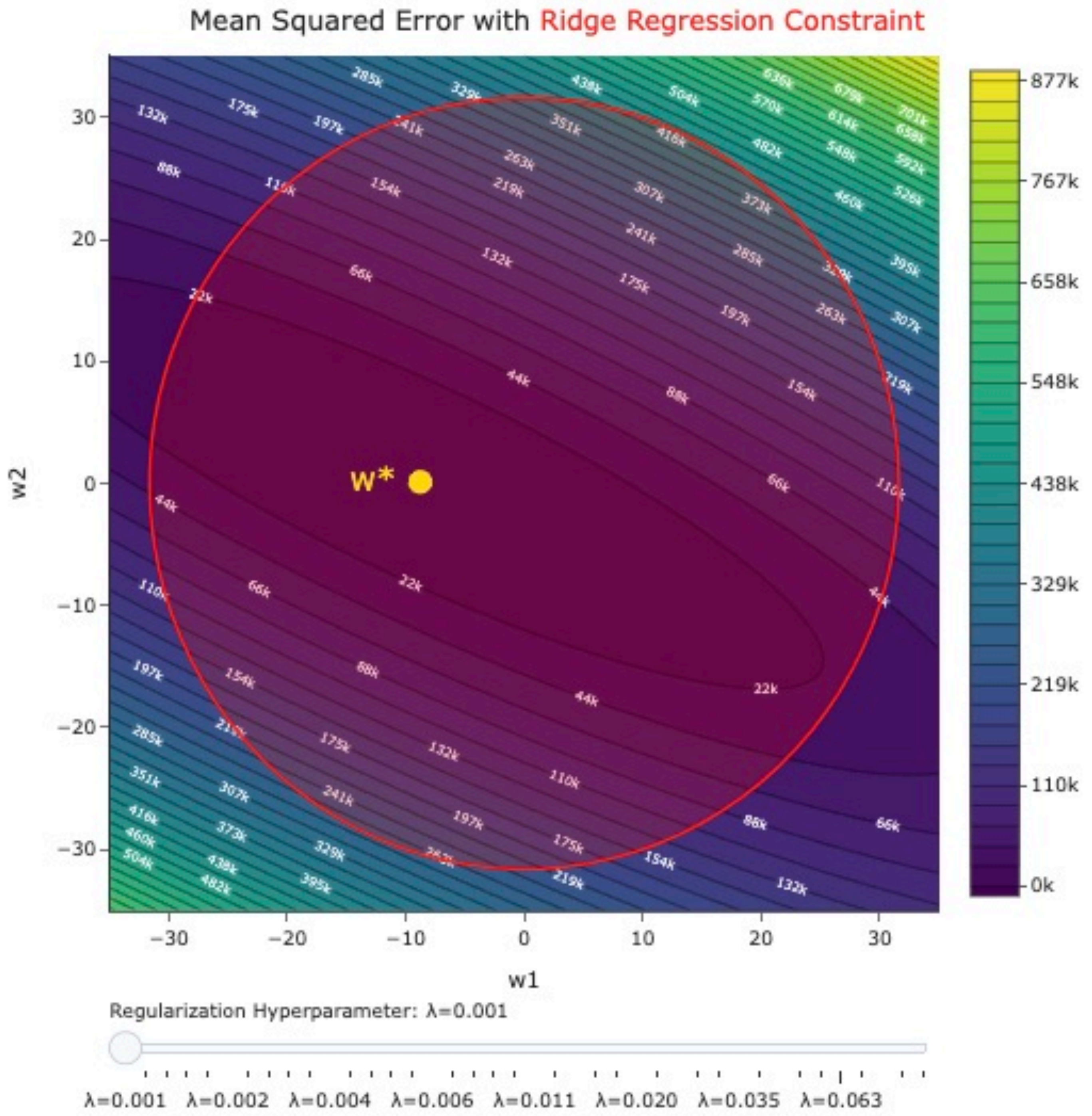
$$\text{minimize} \quad \frac{1}{n}\|\vec{y} - X\vec{w}\|^2 \text{ such that } \sum_{j=1}^{d} w_j^2 \leq Q; \qquad \lambda \approx \frac{1}{Q}$$

- Intuitively:
  - The **contour plot of the loss surface** for just the mean squared error component is in **viridis**.
  - The constraint, $\sum_{j=1}^{d} w_j^2 \leq Q$, is in **red**. Ridge regression says, minimize mean squared error, **while staying in the red circle**.

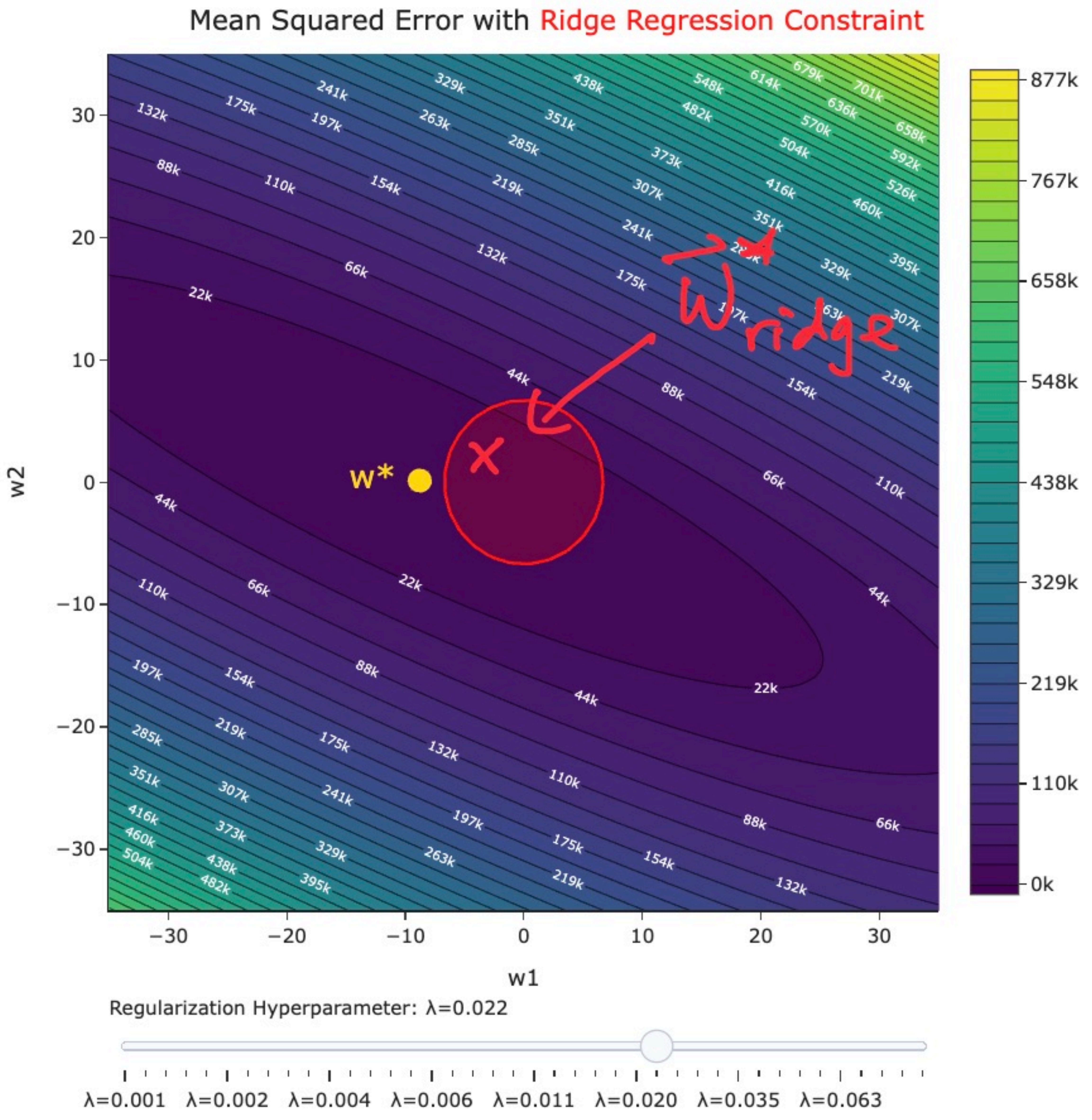
    The larger Q is, the larger the radius of the circle is.

$w_1^2 + w_2^2 \leq 25$

In [9]:
```
util.show_ridge_contour()
```



In [9]:
```
util.show_ridge_contour()
```



Taking a step back

- $\vec{w}^*_{\text{ridge}}$ **doesn't** minimize mean squared error – it minimizes a slightly different objective function.

- So, why would we use ever use ridge regression?

hopefully prevents overfitting.

- What do the **resulting predictions** look like?

```
In [12]: util.plot_given_model_dict(X_train, y_train, {'Extremely Regularized Polynomial of Degree 25': (model_large_lam
```

- What do you notice?

- The L_2 norm, or Euclidean norm, of a vector $\vec{v} \in \mathbb{R}^n$ is defined as:

$$\|\vec{v}\| = \|\vec{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2} = \left(v_1^2 + v_2^2 + \ldots + v_n^2\right)^{\frac{1}{2}}$$

The L_2 norm is the default norm, which is why the subscript 2 is often omitted.

- The L_p norm of $\vec{v}$, for $p \geq 1$, is:

$$\|\vec{v}\|_p = \left(|v_1|^p + |v_2|^p + \ldots + |v_n|^p\right)^{\frac{1}{p}}$$

30 . 1

$$\|\vec{v}\| = \|\vec{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2} = \left(v_1^2 + v_2^2 + \ldots + v_n^2\right)^{\frac{1}{2}}$$

The L_2 norm is the default norm, which is why the subscript 2 is often omitted.

- The L_p norm of $\vec{v}$, for $p \geq 1$, is:

$$\|\vec{v}\|_p = \left(|v_1|^p + |v_2|^p + \ldots + |v_n|^p\right)^{\frac{1}{p}}$$

- Ridge regression is said to use L_2 regularization because it penalizes the (squared) L_2 norm of $\vec{w}$, ignoring the intercept term:

$$R_{\text{ridge}}(\vec{w}) = \frac{1}{n}\|\vec{y} - X\vec{w}\|^2 + \lambda \sum_{j=1}^{d} w_j^2$$

- LASSO is said to use L_1 regularization because it penalizes the L_1 norm of $\vec{w}$, ignoring the intercept term:

$$R_{\text{LASSO}}(\vec{w}) = \frac{1}{n}\|\vec{y} - X\vec{w}\|^2 + \lambda \sum_{j=1}^{d} |w_j|$$

- ## Let's use LASSO to fit a degree 25 polynomial to Sample 1.

Here, we'll **fix** the degree, and cross-validate to find λ.

In [34]:

```python
hyperparams = {
    'lasso__alpha': 10.0 ** np.arange(-2, 15)
}
model_regularized_lasso = GridSearchCV(
    estimator=make_pipeline(PolynomialFeatures(25, include_bias=False), Lasso()),
    param_grid=hyperparams,
    scoring='neg_mean_squared_error'
)
model_regularized_lasso.fit(X_train, y_train)
```

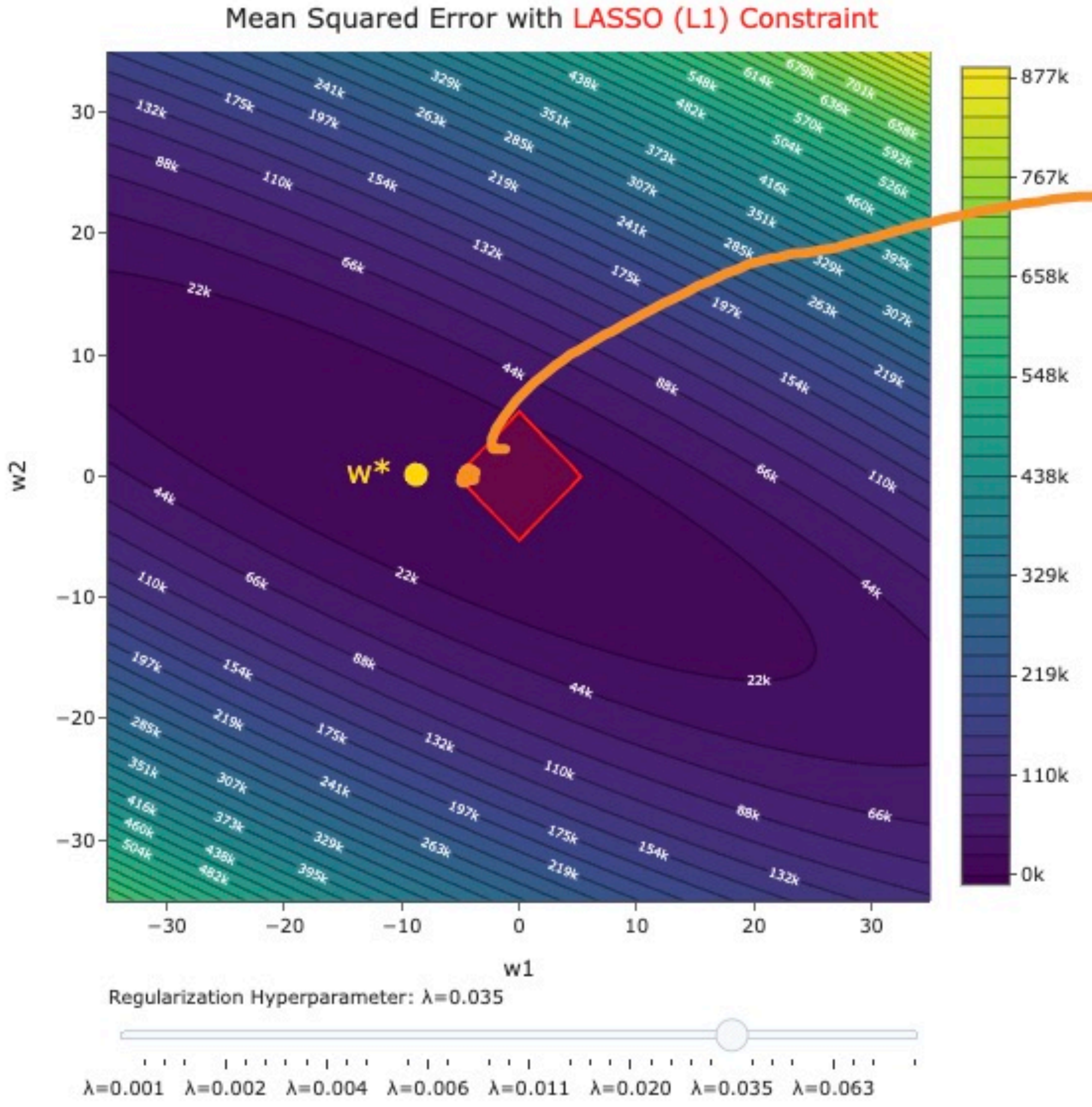
Out[34]:

```
▸ GridSearchCV
▸ best_estimator_: Pipeline
    ▸ PolynomialFeatures
    ▸ Lasso
```

- As before:
  - The **contour plot of the loss surface** for just the mean squared error component is in **viridis**.
  - The constraint, $\sum_{j=1}^{d} |w_j| \leq Q$, is in **red**. LASSO says, minimize mean squared error, **while staying in the red diamond**.

    The larger Q is, the larger the side length of the diamond is.

In [42]:
```
util.show_lasso_contour()
```

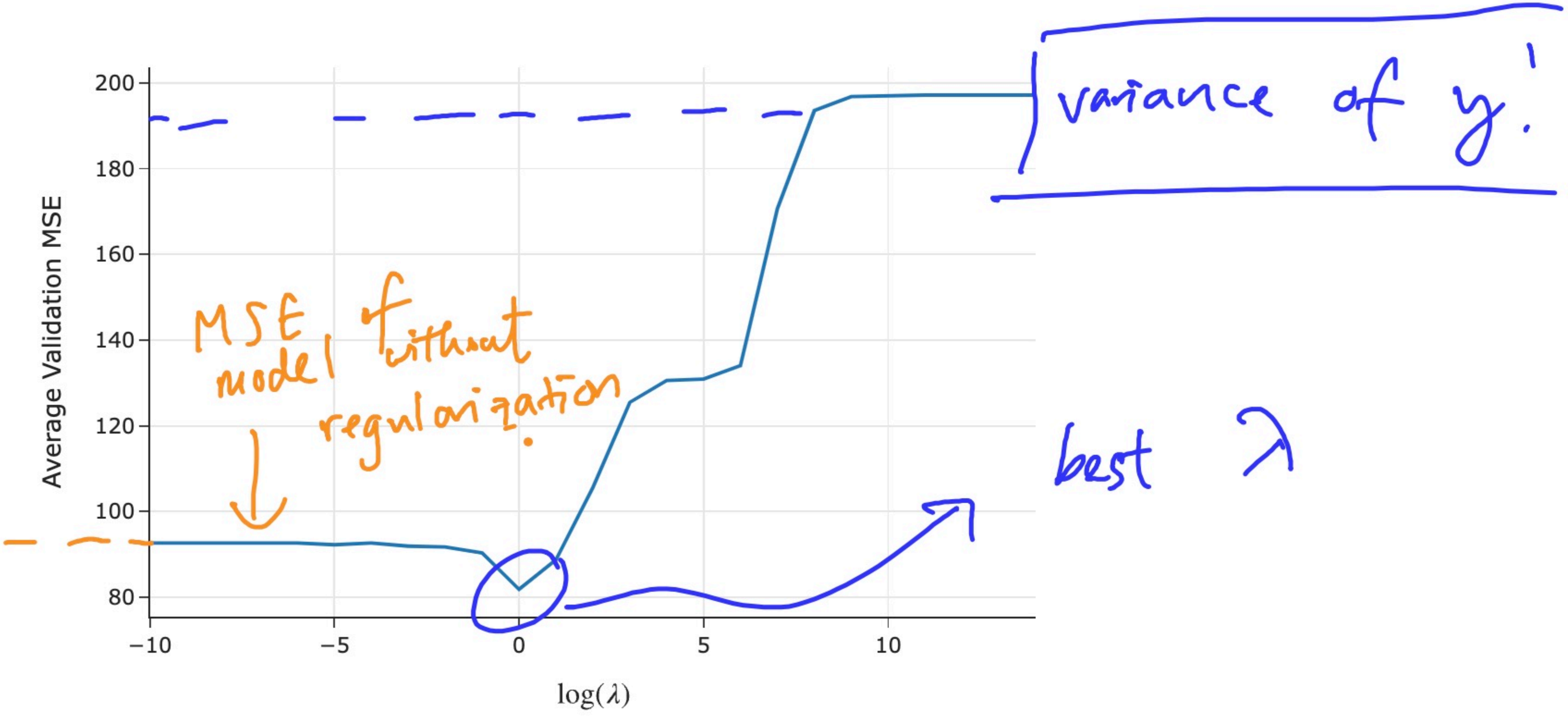


often, the $\vec{w}$ within

the diamond

that minimizes MSE is at a corner.

```
    .update_layout(xaxis_title='$\log(\lambda)$', yaxis_title='Averag
)
```



	ols	ridge	lasso
feature			
intercept	460.31	214.15	2.54e+02
polynomialfeatures__departure_hour	-94.79	-0.71	-2.10e+01
polynomialfeatures__departure_hour^2	6.80	-4.63	-1.70e+00
polynomialfeatures__departure_hour^3	-0.14	0.31	1.81e-01
onehotencoder__day_Mon	-0.61	-5.74	-2.70e+00
onehotencoder__day_Thu	13.30	6.04	9.00e+00
onehotencoder__day_Tue	11.19	5.52	8.68e+00
onehotencoder__day_Wed	5.73	-0.46	0.00e+00
onehotencoder__month_December	8.90	2.82	4.06e+00
onehotencoder__month_February	-5.33	-7.14	-5.81e+00
onehotencoder__month_January	1.93	0.39	0.00e+00
onehotencoder__month_July	2.46	0.44	0.00e+00
onehotencoder__month_June	6.28	4.45	5.14e+00
onehotencoder__month_March	-0.76	-1.70	-8.17e-01
onehotencoder__month_May	9.36	4.95	5.57e+00
onehotencoder__month_November	1.40	-1.81	-0.00e+00
onehotencoder__month_October	2.06	0.22	0.00e+00
onehotencoder__month_September	-3.20	0.05	-0.00e+00
pipeline__day_of_month_Week 2	0.91	1.39	3.23e-01
pipeline__day_of_month_Week 3	6.30	4.70	4.57e+00
pipeline__day_of_month_Week 4	0.28	-0.20	-0.00e+00
pipeline__day_of_month_Week 5	2.09	0.76	4.78e-03

lots of 0s !